Design, Simulation and Virtual Testing

madymo

Coupling Manual | VERSION 7.5 pl1

www.tassinternational.com

© Copyright 2013 by TASS
All rights reserved.

MADYMO® has been developed at TASS BV.

This document contains proprietary and confidential information of TASS. The contents of this document may not be disclosed to third parties, copied or duplicated in any form, in whole or in part, without prior written permission of TASS.

The terms and conditions governing the license of MADYMO® software consist solely of those set forth in the written contracts between TASS or TASS authorised third parties and its customers. The software may only be used or copied in accordance with the terms of these contracts.

MADYMO Manuals

An overview of the MADYMO solver related manuals is given below. From Acrobat Reader, these manuals can be accessed directly by clicking the manual in the table below. Manuals marked with a star (*) are also provided in hard-copy (major releases only).

Theory Manual	The theoretical concepts of the MADYMO solver.
Reference Manual*	Detailed information on how to use the MADYMO solver and how to specify the input.
Model Manual*	Dummy, Dummy Subsystem and Barrier Models with simple examples.
Applications Manual	Example applications using Dummy, Dummy Subsystem and Barrier Models.
Human Model Manual	Human Models and applications that make use of Human Models.
Tyre Model Manual	Documentation about Tyre Models.
Utilities Manual	User's guide for MADYMO/Optimiser, MADYMO/Scaler, MADYMO/Dummy Generator, MADYMO/Tank Test Analysis
Folder Manual	Describes the use of MADYMO/Folder.
Programmer's Manual	Information about user-defined routines.
Release Notes	Describes the new features, modifications and bug fixes with respect to the previous release.
Installation Instructions	Description for the system administrator to install MADYMO.
Coupling Manual	Description of the direct coupling with ABAQUS, LS-DYNA, PAM CRASH/SAFE and Radioss and the TCP/IP coupling with MATLAB/Simulink.

TASS provides extensive and high quality support for its products to help you in utilizing the software most efficiently. TASS offers extensive hotline support for our software products, MADYMO, PreScan and Delft-Tyre. Our hotline support can be reached over phone as well as via email and will assist you with your questions regarding our different software products. Your requests will be dealt with in a fast and effective manner to support you in the continuation of your work in progress. On the website you will find your local representative with the accompanying support contact details.

Contents

MADYMO Manuals iii

1 General Information 1

1.1 Introduction . . . 1

1.1.1 Why Coupling? . . . 1

1.1.2 Coupling Type . . . 1

1.1.3 Simulation Control . . . 3

1.2 Platforms and versions . . . 5

1.3 Program Installation . . . 5

1.4 Performance Guidelines . . . 7

1.5 Limitations and Known Issues . . . 8

2 Interface Version 10

2.1 Introduction . . . 10

2.2 Features of the 6.3.1 interface and onwards . . . 10

2.2.1 Identifing the Interface . . . 10

2.3 Interface History . . . 10

3 Madymo Input Deck Setup 12

3.1 MADYMO input deck setup for Basic Coupling . . . 12

3.2 MADYMO input deck setup for Extended Coupling . . . 13

3.3 Troubleshooting . . . 16

4 Coupling with ABAQUS 18

5 Coupling with LS-DYNA 19

5.1 Introduction . . . 19

5.2 Simulation Control . . . 19

5.3 LS-DYNA Input Deck Setup . . . 19

5.3.1 Basic Coupling . . . 20

5.3.2 Extended Coupling . . . 21
5.4 Coupling on Microsoft Windows platform . . . 23
5.5 Restart . . . 23
5.5.1 Restart Control . . . 24
5.6 Limitations and Known Issues for indirect coupling . . . 24

6 Coupling with PAM CRASH/SAFE **26**
6.1 Introduction . . . 26
6.2 Simulation Control . . . 26
6.3 PAM CRASH/SAFE Input Deck Setup . . . 26
6.3.1 Basic Coupling . . . 26
6.3.2 Extended Coupling . . . 28
6.4 Limitations and Known Issues . . . 30

7 Coupling with Radioss **31**
7.1 Introduction . . . 31
7.2 Simulation Control . . . 31
7.3 Radioss Input Deck Setup . . . 32
7.3.1 Basic Coupling . . . 32
7.3.2 Extended Coupling . . . 33
7.4 Restart . . . 35
7.4.1 Restart Control . . . 35
7.5 Limitations and Known Issues . . . 35

8 Example Startup Scripts **37**
8.1 Direct coupling . . . 37
8.1.1 PAM CRASH/SAFE . . . 37
8.1.2 LS-DYNA . . . 37
8.1.3 MPP-DYNA . . . 38
8.1.4 Radioss . . . 39
8.2 MPP coupling . . . 40
8.2.1 Introduction . . . 40

8.2.2 What does it do . . . 40
8.2.3 Usage . . . 41
8.2.4 Hostfiles . . . 42
8.2.5 Configuration Files . . . 43
8.2.6 Argument Interface . . . 44
8.2.7 Checklist and Example . . . 44
8.2.8 Restart . . . 45
8.2.9 Example wrapper scripts . . . 45
8.2.10 Troubleshooting . . . 49

9 MATLAB/Simulink coupling **51**

9.1 Compatible MATLAB version . . . 51
9.2 Interfacing MADYMO with MATLAB/ Simulink . . . 52
9.3 Using MATLAB/ Simulink in combination with MADYMO . . . 53
9.4 Conditions . . . 55
9.5 Miscellaneous . . . 55

1 General Information

1.1 Introduction

This manual gives an overview of the requirements to perform a simulation with MADYMO coupled to another program. Additionally, it explains why MADYMO facilitates a coupling with other software, how the coupling works and what types of coupling are supported. Hereafter, the coupling to another program is generically termed PARTNER, unless otherwise specifically indicated. All information in this section applies to all partners unless otherwise specified.

1.1.1 Why Coupling?

A coupling to another software product provides the opportunity for the user to combine, in a single simulation, the features from both products. The user may need to combine the two products because some features may be available in one product and not in the other or the user may simply rely on one software product for one type of calculation and on the other product for other types of calculations.

1.1.2 Coupling Type

The coupling between MADYMO and the PARTNER can be done by including MADYMO as a library in the PARTNER, called direct coupling, and by keeping MADYMO and PARTNER separated. The latter principal is used in the MPP coupling.

Direct coupling

The coupling between two software products is done through a coupling interface. The information to be send from one software product to the other is first gathered internally. This information is then send to the PARTNER through the coupling interface routines. The PARTNER distributes the data, performs calculations and subsequently gathers data and sends is back through the interface. This process is illustrated in Fig. 1.1.

MPP coupling

The coupling between two software products is done through the coupling library MADCL. The information to be send from one software product to the other is not gathered and distributed but instead all MPP processes on MADYMO side can communicate directly to all MPP processes on the PARTNER side. The coupling library MADCL takes care of the administration, i.e. which process can (and should) talk (and listen) to which other process.

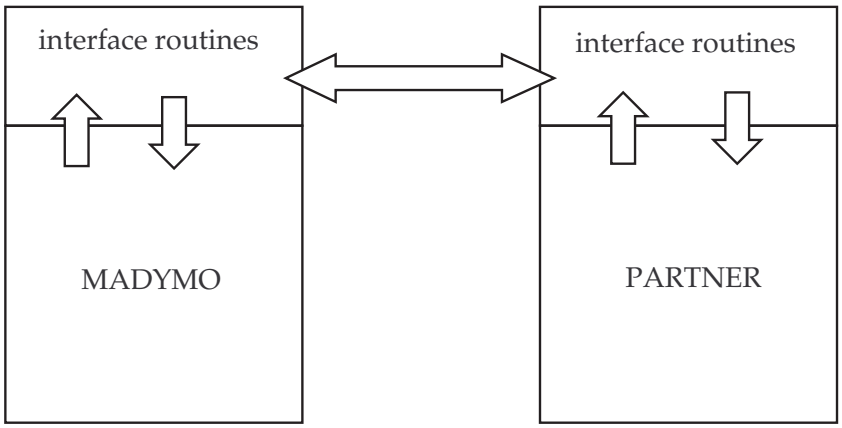


Figure 1.1: Illustration of the communication through the coupling interface.

Basic versus Extended Coupling

The Direct coupling and the MPP coupling differ in what type of data is communicated between MADYMO and the PARTNER. Precisely, the direct coupling supports both basic coupling as well as extended coupling and combinations thereof. MPP coupling only supports extended coupling. Basic and Extended coupling are defined as follows:

- MB object data is communicated; at each time integration point, displacements are sent from MADYMO to the PARTNER code, which in turn sends forces acting upon these objects back to MADYMO. This is called *Basic Coupling*.

  In basic coupling, the PARTNER program builds a copy of the MB object natively in MADYMO. This copy allows the PARTNER program to put forces acting on the MB object, and to display it in its kinematic output file.

- FE data is communicated; at each time integration point, nodal displacements are sent from the PARTNER program to MADYMO, which in turn sends forces acting on these nodes back to the PARTNER program. This is called the *Extended Coupling*.

  In extended coupling, MADYMO builds a copy of (part of) the FE model it receives from the PARTNER program. Now MADYMO can put forces on the nodes, and display the FE model in the kinematic output file (KIN3 file).

- It is also possible to mix these two couplings. This is called *Combined Coupling*.

For direct coupling the reprint file reflects the coupling type under the - - - COUPLING - - - section.

All coupling types synchronize their time step, so the minimum of the PARTNER and MADYMO time step is taken.

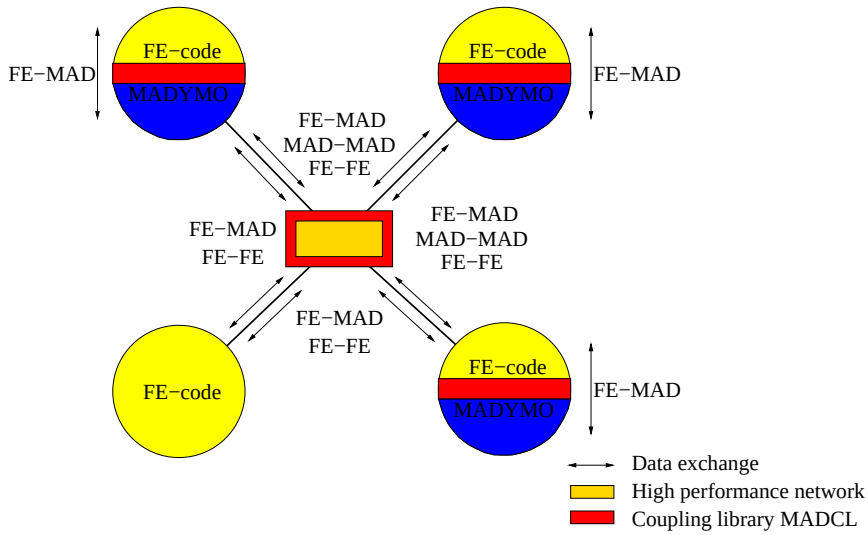


Figure 1.2: Illustration of the communication through the coupling library MADCL.

When MADYMO dummy models are used in coupling it is recommended to use the extended coupling. This is because the compliance of the dummy contact surfaces is not taken into account in the basic coupling. However, the basic coupling generally runs more efficiently, because less data needs to be communicated between MADYMO and the PARTNER.

Please be aware that this MADYMO release supports both basic and extended coupling for direct coupling. The direct coupling however may not be supported by newer versions of the PARTNER codes. Details for direct coupling are given in Tab. 1.1–1.4.

MPP coupling is supported by newer versions of the PARTNER codes. Details are given in Tab. 1.5.

1.1.3 Simulation Control

- The simulation end time has to be defined both in PARTNER and MADYMO. When defined differently in both, the simulation will end when the lowest end time is reached.
- The kinematic output time step in the MADYMO input file and the parameters of the option in the PARTNER input file must be consistent. The combined output will be out of phase, otherwise.
- For Direct Coupling define the number of CPU's in the MADYMO input deck. If you plan to run an SMP job, the number of CPU's used by MADYMO and PARTNER must be equal. MADYMO defines the number of CPUs, say #n, by NR_PROC in the XML input file with

MADYMO		LS-Dyna	
Platform id	last release	**970**	**971** up to **R3.1**
hp1100pa20	R7.1	**NT**	**R** [1 Dec 2008]
hp1100ia64	R7.4	**NT**	**R** [1 Dec 2008]
sgi64r10k	R7.1	**NT**	**R** [1 Dec 2008]
ibmrs51	R7.4	**NT**	**U**
linux24-x86	R7.4.1	**NT**	**R** [1 Dec 2008]
linux24-ia64	R7.4	**NT**	**U**
linux26-x86_64		**X** [Compiler Conflict]	**R** [1 Dec 2008]
linux24-x86_64	R7.3	**X** [Compiler Conflict]	**R** [1 Dec 2008]
linux24-em64t	R7.3	**X** [Compiler Conflict]	**CV** [linux24-x86_64]
win32		**X** [security issue]	**X** [security issue]
em64t-win		**X** [security issue]	**X** [security issue]

Table 1.1: Direct coupling support for LS-Dyna in R7.5.

MADYMO		MPP-Dyna	
Platform id	last release	**970**	**971** up to **R3.1**
hp1100pa20	R7.1	**R** [1 Dec 2008]	**U**
hp1100ia64	R7.4	**V** [1 Dec 2008]	**U**
sgi64r10k	R7.1	**U**	**U**
ibmrs51	R7.4	**U**	**U**
linux24-x86	R7.4.1	**V** [1 Dec 2008]	**U**
linux24-ia64	R7.4	**V** [1 Dec 2008]	**V** [1 Dec 2008]
linux26-x86_64		**V** [1 Dec 2008]	**V** [1 Dec 2008]
linux24-x86_64	R7.3	**V** [1 Dec 2008]	**V** [1 Dec 2008]
linux24-em64t	R7.3	**CV** [linux24-x86_64]	**CV** [linux24-x86_64]
win32		**X** [security issue]	**X** [security issue]
em64t-win		**X** [security issue]	**X** [security issue]

Table 1.2: Direct coupling support for MPP-Dyna in R7.5.

```
<CONTROL_ALLOCATION
    NR_PROC  = "#n"
/>
```

For MPP Coupling, the number of CPU's for MADYMO and PARTNER is independent of each other and must be specified on the command line in accordance with starting up a stand-alone MADYMO MPP calculation. Note that a start-up script `madymo_cpl` is available to aid in starting up an MPP Coupling calculation, see Sec. 8 for more details.

- Generally, for Direct Coupling command line options cannot be used. For example, -nrproc (to set the number of CPU's), -isize, -rsize, -csize (to set memory allocation) and

MADYMO		Pam Crash	
Platform id	last release	**2007 SMP**	**2007 MPP**
hp1100pa20	R7.1	**?**	**?**
hp1100ia64	R7.4	**R** [1 Dec 2008]	**R** [1 Dec 2008]
sgi64r10k	R7.1	**?**	**?**
ibmrs51	R7.4	**?**	**?**
linux24-x86	R7.4.1	**R** [1 Dec 2008]	**R** [1 Dec 2008]
linux24-ia64	R7.4	**R** [1 Dec 2008]	**R** [1 Dec 2008]
linux26-x86_64		**R** [1 Dec 2008]	**R** [1 Dec 2008]
linux24-x86_64	R7.3	**R** [1 Dec 2008]	**R** [1 Dec 2008]
linux24-em64t	R7.3	**R** [1 Dec 2008]	**R** [1 Dec 2008]
win32		**X** [security issue]	**X** [security issue]
em64t-win		**X** [security issue]	**X** [security issue]

Table 1.3: Direct coupling support for Pam Crash in R7.5.

others cannot be used. Please use the input decks to specify the required information.[1]
For MPP Coupling, please refer to the start-up script `madymo_cpl`.

1.2 Platforms and versions

In order to run a direct coupling simulation, one needs both the special coupling executable as well as the MADYMO shared library. Starting from MADYMO R6.3, the latter is distributed by default with each MADYMO release/patch on all supported Unix and Linux platforms (please refer to the MADYMO Installation Instructions). The special executable from the PARTNER software may be obtained from the PARTNER offices.

With the introduction of the MADYMO shared library, the user can choose which MADYMO version he wants to use with a certain PARTNER version, and vice versa. To do so, he simply uses a newer version of MADYMO. If set up correctly, the PARTNER executable will pick up the new shared library.

For a list of validated MADYMO–PARTNER combinations, please refer to Tab. 1.1–1.4. Please refer to the MADYMO Installation Instructions for an up to date list of supported platforms.

1.3 Program Installation

This section describes the steps needed for installation in order to perform a coupling simulation.

[1] There is one exception: specifying the number of CPU's when coupling with PAM CRASH/SAFE, see Sec. 8.1.1

MADYMO		Radioss	ABAQUS
Platform id	last release	**5.1 I**	**6.7 PR3E**
hp1100pa20	R7.1	**?**	**?**
hp1100ia64	R7.4	**?**	**V** [1 Dec 2008]
sgi64r10k	R7.1	**?**	**?**
ibmrs51	R7.4	**?**	**?**
linux24-x86	R7.4.1	**?**	**?**
linux24-ia64	R7.4	**?**	**V** [1 Dec 2008]
linux26-x86_64		**R** [1 Dec 2008]	**V** [1 Dec 2008]
linux24-x86_64	R7.3	**R** [1 Dec 2008]	**V** [1 Dec 2008]
linux24-em64t	R7.3	**CV** [linux24-x86_64]	**V** [1 Dec 2008]
win32		**X** [security issue]	**X** [security issue]
em64t-win		**X** [security issue]	**X** [security issue]

Table 1.4: Direct coupling support for Radioss and Abaqus in R7.5.

- **?** - Not Requested: Platform is not requested by customers.
- **U** - Under Development: Agreement between MADYMO and PARTNER to start development. No Committment yet to deliver this version.
- **NT** - Not Tested yet: The platform needs to be tested, but was validated with older versions. Will be done on customer request.
- **V** - Validated: Validated by MADYMO (and optionally PARTNER) but not yet released by partner (date of validation mentioned).
- **R** - Released: Validated by MADYMO/PARTNER and (officially) released by PARTNER (date of release by partner mentioned).
- **X** - Not Possible: Not possible to create a version due to technical reasons (reason mentioned).
- **CV** - Compatible Version: Compatible version available (replacement platform mentioned).

	Pam Crash	**LS-Dyna, MPP-Dyna**	**Radioss**
	Available upon request from ESI	971 R5.1.1 and later	V10.0.0 and later
linux26-x86_64	**R**	**R**	**R**
win32	**?**	**R**	**?**
em64t-win	**?**	**R**	**?**

Table 1.5: MPP coupling support for R7.5.

1. Install MADYMO according to the MADYMO Installation Instructions. If MADYMO is already installed, go to step 3.
2. Install the MADYMO license file *madymo.lic* for MADYMO.
3. Verify that the MADYMO license for coupling to the PARTNER is present in the MADYMO license file.
4. Install PARTNER software according to their installation instructions, including, if needed, licenses required by PARTNER for using the coupling with MADYMO.
5. For direct coupling, install the special coupling executable which is to be obtained from the PARTNER. For MPP coupling no additional coupling executable is needed.
6. Add the path to the **madymo75** run script to the PATH environment.

1.4 Performance Guidelines

Starting from MADYMO 6.3, all MADYMO shared libraries are SMP. This implies that all direct coupling simulations are SMP enabled. For SMP-SMP calculations, in general more CPU's will result in a faster runtime (up to approx 4 to 8 CPU's, depending on the speedup of the PARTNER). The performance of an SMP-MPP calculation depends both on the machine(s) on which the job is run and on the input deck. For the SMP-MPP coupling it is strongly recommended to run MADYMO on 1 CPU.

Coupling simulations in which MADYMO runs in MPP mode are only supported for MPP coupling.

- The amount of data that is to be communicated, invariably has an impact on the performance. Besides being communicated twice, the data is also gathered and distributed twice every time step in direct coupling. To increase performance, please have a close look at the data that is communicated and remove any unneeded or unused data.
- The user is advised to check the MADYMO Timing Information in the log and/or reprint file. It will give an indication of the CPU ratio that each party takes in the coupling calculation. MADYMO typically takes 5 - 15% of the CPU time. Increasing MADYMO performance will not give good over-all speed improvements. For example, when MADYMO uses 10% of the CPU time, even halving the MADYMO CPU time will only yield an overall speed increase of 5%. Significant speed improvements are generally to be found in the party which uses most of the CPU time.

  In MPP coupling, the data to be communicated is not gathered and distributed (depending on the implementation by the PARTNER). Instead the communication between processes from MADYMO and PARTNER is done directly between the processes, facilitated by the coupling library MADCL.
- Especially for SMP-MPP calculations, have a look at the CPU/Wall Clock ratio. In general, this should be higher than 90%. Low ratios can be caused by (among others)

  - Allocating more nodes than available CPU's.
  - Bad network connections
  - Wrong MPI version (especially for Linux)
  - Sending large amounts of data over the coupling interface

  If in doubt, please contact your local support office. For direct coupling, note that MADYMO does not use any MPI calls, so any problems caused by it can best be handled by the PARTNER support office. For MPP coupling the logging of MADYMO and PARTNER are separated. When MPP coupling calculations fail, first check the available logging and contact the support office of the party that aborts the calculation.

- When the user is in doubt if the coupling is having a bad impact on the performance, please run the coupled executable as if it was a stand-alone executable. All coupled executables provided by PARTNER, can be run in stand-alone mode. By disabling coupling (both in keywords in the input deck and command line options), all influences of the coupling interface are removed and the most objective comparison between coupling and standalone can be made. Please keep in mind that for best comparison, all MADYMO data must be replaced by the PARTNER counterparts.

1.5 Limitations and Known Issues

- Restarting a coupled simulation is supported with Radioss and LS-DYNA in direct coupling. MPP coupling supports restart for LS-DYNA and MPP-DYNA, the latter with madymo running on one cpu.
- In direct coupling, the PARTNER program sometimes writes "NORMAL TERMINATION", even when MADYMO terminates abnormally.

  Therefore the user is advised to check the following messages after running a coupled simulation:

  - In the screen output[2], PARTNER writes "NORMAL TERMINATION" (or a similar message) if no error appears on the PARTNER side.
  - The message: "MADYMO TERMINATED NORMALLY" must appear in the log file and reprint file.
- For extended coupling, it is required to send at least one element from PARTNER to MADYMO. It is not allowed to send only nodes.
- Planes attached to a body are incorrect positioned if the center of gravity is nonzero.
- When performing an SMP-MPP (direct coupling) job, the output to standard out (typically the screen), and therefore the MADYMO log file, can be out of phase. This is normal behavior. On some systems, this behavior can be changed, see the MPI documentation for more information.

[2]Not all partners write there output such that it ends up in the log file

- On AMD Opteron systems (linux24-x86_64 and linux26-x86_64), MADYMO R6.3.1 and higher are incompatible with PARTNER executables that run with MADYMO R6.3. The job stops with the following error:

  ```
  <cplexe>: error while loading shared libraries: <cplexe>: undefined symbol:
  pghpf_sect1
  ```

  where <cplexe> is the PARTNER coupling executable.

  The solution is to update the PARTNER executable to a version build with PGI compiler version 6.1.

  The MPP coupling is designed to avoid compiler dependencies between MADYMO and PARTNER.

2 Interface Version

2.1 Introduction

With the introduction of MADYMO 6.3.1, a coupling interface is used. This interface has new features, yet provides full backwards compatibility to enable seamless integration with older PARTNER executables that do not have this interface build-in.

2.2 Features of the 6.3.1 interface and onwards

The interface provides two features:

Automatic Kinematic Output Scaling MADYMO can scale its kinematic output to the length units of the PARTNER. To enable this, set the AUTO_SCALE_ANI keyword under COUPLING to "ON". The default behaviour is "OFF", to enable backwards compatibility.

Communication of failure exit to PARTNER codes MADYMO sends an error termination flag to the PARTNER code, such that it can write an abnormal termination message to the screen. Also, the PARTNER code can send such a flag to MADYMO, causing MADYMO to write an abnormal termination message to the log and reprint file.

For both features applies that the PARTNER executable must support this, meaning that they must also use the interface.

2.2.1 Identifing the Interface

When MADYMO notices that the PARTNER executable is using an older interface, MADYMO will not use these features and fall back on a version that the PARTNER software does support. In this way, the largest common version of the interface is used. This version is the *effective interface version*. The reprint file reflects this version under the — COUPLING INTERFACE — section.

When this *effective interface version* is not reflected in the reprint file, an older version of MADYMO (version 6.3 or earlier) is used. These versions are always build with interface version 0, and therefore do not have the abovementioned improvements.

2.3 Interface History

This section reflects what MADYMO releases are build with which interface versions, newest first. For a detailed list of supported and tested platforms, consult the MADYMO public download area at www.tassinternational.com.

Direct Coupling:

Version	Release	Description
1	6.3.1 and onwards	Automatic Kinematic Output Scaling Error Termination Flag
0	6.3	Interface as used with MADYMO version 6.3

MPP Coupling:

Version	Release	Description
0.9	R7.2pl2	First released version of coupling library madcl.
1.0	R7.3pl1	LS-DYNA windows coupling enabled.

3 Madymo Input Deck Setup

In order to enable coupling to an external executable, the MADYMO input deck must contain the <COUPLING/> element. Note that one must always select INT_MTH = "EULER" under CONTROL_ANALYSIS.TIME for a coupling simulation.

3.1 MADYMO input deck setup for Basic Coupling

This section describes the elements that must be added to the MADYMO input deck for a Basic Coupling simulation.

Note that basic coupling is not supported in the coupling with ABAQUS.

- Rigid bodies which are to be coupled to PARTNER are referred to under the COUPLING element as shown in the following example:

```
<COUPLING>
  <COUPLING_BODY
    EXTERNAL_REF ="1"
    BODY ="/1/1"
    EXTERNAL_DATA ="163"
  />
</COUPLING>
```

  - The values in EXTERNAL_REF and EXTERNAL_DATA are send to PARTNER. For more details, please refer to Sec. 6.3.1 for basic coupling to PAM CRASH/SAFE or Sec. 7.3.1 for basic coupling to Radioss.
  - Bodies cannot be coupled to LS-DYNA.

- Ellipsoids and planes which are to be coupled to PARTNER are defined under the COUPLING element as shown in the following example:

```
<COUPLING>
  <COUPLING_SURFACE
    EXTERNAL_REF="3"
    SURFACE="/2/1"
    EXTERNAL_DATA="9"
  />
</COUPLING>
```

  - The values in EXTERNAL_REF and EXTERNAL_DATA are send to PARTNER. For more details, please refer to Sec. 5.3.1 for basic coupling to LS-DYNA, Sec. 6.3.1 for basic coupling to PAM CRASH/SAFE or Sec. 7.3.1 for basic coupling to Radioss.
  - Cylinders cannot be coupled to LS-DYNA, PAM CRASH/SAFE and Radioss.

The elements <COUPLING_BODY/> and <COUPLING_SURFACE/> may be combined within the same <COUPLING/> element.

3.2 MADYMO input deck setup for Extended Coupling

This section describes the XML elements that should be added to the MADYMO input deck for an Extended Coupling simulation.

When performing an extended coupling simulation, the amount of data transferred between PARTNER and MADYMO can make it necessary to increase the double precision and integer space in MADYMO, depending on the model sizes. Therefore, the user may have to increase the double precision and integer space of MADYMO using the I_SIZE and R_SIZE settings in the MADYMO XML input deck under CONTROL_ALLOCATION.

In the unlikely event that MADYMO complains about the character space allocated, also increase the C_SIZE setting.

It is recommended to perform a so called 'zero-run' first (with the end time identical to the start time). Inspect the MADYMO reprint file (between the lines COUPLING_INTERFACE and END COUPLING_INTERFACE) to check that MADYMO reads the data correctly from PARTNER. Also check any errors and warnings reported by either MADYMO or PARTNER.

Topology of (degenerated) elements can be transferred from PARTNER to MADYMO. Elements of type TRIAD3, QUAD 4 and HEXA8 are allowed to be sent over.

1. Add an empty FE_MODEL to the MADYMO input file (under SYSTEM.*) to represent the PARTNER model that will be coupled to MADYMO. Within MADYMO, all data received from the PARTNER is assembled in a single FE model. For example:

```
<SYSTEM.MODEL
   ID="2"
   NAME="Ext_System"
   >
   <FE_MODEL
      ID="1"
      NAME="Ext_FE_Model"
      >
      <CONTROL_FE_MODEL
      />
      <CONTROL_FE_TIME_STEP
      />
   </FE_MODEL>
</SYSTEM.MODEL>
```

2. Add a COUPLING element with a reference to this FE model:

```
<COUPLING
   FE_MODEL="/Ext_System/Ext_FE_Model"
/>
```

3. With the previous steps, the MADYMO side has been prepared to receive data from the external FE-model. The coupled FE model can be used similar to a native MADYMO FE model, with the restriction that no kinematic constraints can be used. Specifically, kinematic contact, spot welds, prescribed motion and rigid elements cannot be defined in the MADYMO input deck for the external FE model. Next, the user must define what to do with this data. Four examples follow.

(a) Contact Interaction

If an external group is to be used in a contact calculation, add the following:

i. Define a specific GROUP_FE elements for the PARTNER elements involved in the contact interaction. Groups of elements, nodes and/or parts can be used.

```
<GROUP_FE
   ID="2"
   NAME="Ext_Group_gfe"
   FE_MODEL  ="/Ext_System/Ext_FE_Model"
   PART_LIST ="ALL"
/>
```

The attribute PART_LIST may also be combined or replaced by other lists, for example a NODE_LIST and/or ELEMENT_LIST (see the MADYMO Reference Manual for details). The value ALL may also be replaced by more specific part, node or element numbers provided that the PARTNER input deck specifies that these values are communicated to MADYMO. Please note that numbering of nodes and elements of the PARTNER FE coupled parts in MADYMO may not be the same as in the PARTNER FE input deck.

ii. Define a contact between this group and the MADYMO model groups using CONTACT.FE_FE or CONTACT.MB_FE. Here follow two examples:

```
<CONTACT.FE_FE
      ID="21"
      MASTER_SURFACE="/ES2_lhs/Pelvis_gfe"
      SLAVE_SURFACE="RADIOSS_door_gfe"
      >
      <CONTACT_METHOD.NODE_TO_SURFACE_CHAR>
         <CONTACT_FORCE.CHAR
            CONTACT_TYPE="MASTER"
            FRIC_FUNC="Friction03_fun"
         />
      </CONTACT_METHOD.NODE_TO_SURFACE_CHAR>
   </CONTACT.FE_FE>
```

and

```
   <CONTACT.FE_FE
      ID="1"
      MASTER_SURFACE="/Ext_Coupling_gfe"
      SLAVE_SURFACE="/DAB_gfe"
      >
      <CONTACT_METHOD.SURFACE_TO_SURFACE>
         <CONTACT_FORCE.ADAPTIVE
            DAMP_COEF="0.05"
            TIME_STEP="2.05E-6"
         />
         <GAP_TYPE.FUNC
            GAP_FUNC="/Gap_fun"
         />
         <CONTACT_EDGE/>
         <INITIAL_TYPE.CHECK/>
      </CONTACT_METHOD.SURFACE_TO_SURFACE>
   </CONTACT.FE_FE>
```

Please note the following:

- All contact types can be used (elastic and penalty based) except kinematic contact.
- Damping in the contact(s) between MADYMO and PARTNER can be used.
- For contact between a PARTNER FE model and MADYMO MB models use CONTACT.MB_FE.

(b) Connections Defining Tied Surfaces

If (part of) a MADYMO FE model is to be connected to (part of) an external FE model, add a tied surface. The constraints between the MADYMO surface and the external surface can be defined using TIED_SURFACE.SPOTWELD, TIED_SURFACE.BREAK_FORCE/STRESS. Groups in the coupled FE model should be used for the MASTER_GROUP_LIST. For the SLAVE_GROUP_LIST, groups in other FE_MODEL elements in the MADYMO input file should be used. The nodes of the slave group will be tied to the PARTNER master surface.

```
<TIED_SURFACE.SPOTWELD
   ID="96"
   NAME="Spotweld_Cpl"
   SLAVE_GROUP_LIST="spotweld_gfe"
   MASTER_GROUP_LIST="Ext_Group_gfe"
   GAP_TYPE="VALUE"
   GAP_VALUE="0.1"
/>
```

(c) Constraint of MADYMO rigid body and external FE node(s)

SUPPORT_RESTRAINT has been added, based on a point-restraint model. In this way external FE nodes can be constrained to MADYMO rigid bodies.

In the following example the nodes 101:103 of the external FE model "Coupling/Vehicle_part" are supported on the MADYMO rigid body "/SYS1/Cube_bod". The stiffness and damping characteristics are computed based on the time-step and the damping characteristic according to the critical damping because no user CHARACTERISTIC.LOAD is specified.

```
<SUPPORT_RESTRAINT
   BODY = "/SYS1/Cube_bod"
   FE_MODEL = "/Coupling/Vehicle_part"
   NODE_LIST = "101:103"
/>
```

(d) Constraint of MB belt to external FE node

POINT_OBJECT.MB being referred from a BELT_SEGMENT now facilitates connecting a belt segment to an external coupled finite element model. Here follows an example which illustrates the usage.

```
<BELT_SEGMENT
   ID="4"
   NAME="belt4_Buc_Part"
   POINT_REF_1="belt2_Buc"
   POINT_REF_2="belt4_Buc"
   CHAR="char_132"
   INITIAL_STRAIN="5.000000E-05"
   ADD_LENGTH="0.0"
/>
<POINT_OBJECT.MB
   ID="3"
   NAME="belt2_Buc"
   FE_MODEL="/Coupling_Sys/Ext_Fem"
   NODE="4"
   />
<POINT_OBJECT.MB
   ID="8"
   NAME="belt4_Buc"
   FE_MODEL="/Seat_Sys/Shoulder_Belt_Fem"
   NODE="77"
/>
```

(e) Constraint of a jet to external FE nodes
FE_CRDSYS under JET.* (except JET.GAS_FLOW_ELEMENT) now facilitates connecting a jet to an external coupled finite element model. An example which illustrates the usage:

```
<SYSTEM.MODEL
  ID="10"
  NAME="coupling_system.model"
>
  <FE_MODEL
    ID="1"
    NAME="coupling_fe_model"
    >
    <CONTROL_FE_MODEL/>
    <CONTROL_FE_TIME_STEP/>
    <FE_CRDSYS.NODE
      ID="1000"
       NAME="fe_jet_sup_external"
       NODE1="8"
       NODE2="9"
       NODE3="5"
     />
.....................
.....................
  </FE_MODEL>
</SYSTEM.MODEL>
.....................
.....................
    <JET.IDELCHIK
        ID="1"
        FE_CRDSYS="/10/1/1000"
        CENTRE="0.19574338 0.19574338 0.19574338"
        OUTFLOW_DIR="1.0 0.0 0.0"
        >
        <JET_SHAPE.RECTANGULAR
          LONG_LENGTH="0.01"
          SHORT_LENGTH="0.01"
          LONG_DIR="0.0 1.0 0.0"
        />
     </JET.IDELCHIK>
```

3.3 Troubleshooting

The user is advised to check the following topics when having trouble in getting the simulation to run properly:

- Look at all warnings and errors, both from MADYMO as well as the PARTNER program.
- Make sure that the data send from the PARTNER program is in SI units. Some partners allow non-SI units to be used. Use the appropriate conversion factors to force sending all data in SI units.
- Is the MPI environment installed correctly and working OK?
- View the kinematic output from both MADYMO and the PARTNER on top of each other. Have a close look at the data that is send to the other program: is it moving simultaneously (and as expected) in both output files? If not, you probably put some kinematic

constraints on the data that is communicated. Identify these constraints and remove them.

4 Coupling with ABAQUS

The coupling executable and a coupling manual for Abaqus coupling can be obtained from your local Abaqus support office. Coupling with Abaqus is in use by dedicated customers.

5 Coupling with LS-DYNA

5.1 Introduction

This section gives specific information related to the coupling with LS-DYNA. All information is applicable to both the SMP as well as the MPP version of LS-DYNA (MPP-DYNA), unless specifically indicated otherwise.

Please note that the details of the input deck for LS-DYNA have been based on LS-DYNA 970 and may be subject to modifications. Always refer to the latest LS-DYNA documentation for the correct syntax.

5.2 Simulation Control

- For general simulation control, please refer to Sec. 1.1.3.
- Defining the number of CPUs/Nodes to run on:
  - For an SMP job, LS-DYNA uses the number of CPUs defined with the NCPU option. The *CONTROL_PARALLEL keyword in the input file is obsolete.
  - For an MPP job, MPP-DYNA, the number of nodes is defined through the MPI command line option (see Sec. 8.1.3).

5.3 LS-DYNA Input Deck Setup

This section describes the required modifications to the LS-DYNA input deck to perform Basic Coupling or Extended Coupling simulations. See the LS-DYNA User Manual for details on the cards not mentioned here specifically. Modifications to the MADYMO input deck are described in Sec. 3.

For all couplings, add the *CONTROL_COUPLING card to set the units relationship between MADYMO and LS-DYNA:

```
*CONTROL_COUPLING
$#  UNLENG    UNTIME    UNFORC    TIMIDL     FLIPX     FLIPY     FLIPZ    SUBCYL
     1000.        1.        1.        0.         0         0         0         0
```

Here the 1000 specifies that LS-DYNA should divide their length values by 1000 before sending them to MADYMO. Length values received from MADYMO will be multiplied by 1000. This control coupling card example shows that the LS-DYNA length units are: mm. The time and force numbers are unchanged with respect to MADYMO (so second's and Newton's).

5.3.1 Basic Coupling

- Define parts using *PART for representing the MADYMO ellipsoids and planes. This has to be repeated for each ellipsoid and plane in MADYMO.
- Define shells for parts representing the MADYMO ellipsoids and planes using *SECTION_SHELL.
- Define rigid body material behavior for parts representing MADYMO ellipsoids and planes using *MAT_RIGID:

```
*MAT_RIGID
$#     MID        RO         E        PR         N    COUPLE         M  RE/ALIAS
```

  - RE/ALIAS must match the EXTERNAL_REF number of the MADYMO ellipsoid or plane in the COUPLING data block (see Sec. 3.1). RE/ALIAS must be unique for all coupling entities.
  - EXTERNAL_DATA is not used by LS-DYNA.
  - COUPLE: coupling flag. Set to 2 to generate a mesh in LS-DYNA for MADYMO ellipsoids and planes and write these in the d3plot output file.
  - M and N are ignored. These parameters are used for compatibility with the MADYMO 5.4.1 coupling.
  - Beware of correct unit conversion scales of R_0 and E values when defining MAT_RIGID.

- Define contacts for the parts representing MADYMO ellipsoids or planes using *CONTACT_ENTITY. Some coupling related cards are:

  - GEOTYP (card1):
    6: Plane
    7: Ellipsoid
  - SO (card2): Contact stiffness type:
    0: Rigid MADYMO ellipsoid or plane
    1: Contact stiffness to be taken from E value in *MAT_RIGID
    2: User-defined force-penetration curve
  - G1 (card5): ID for MADYMO ellipsoid or plane

- Optional: LS-DYNA nodes can be supported on a MADYMO rigid body using *CONSTRAINED_EXTRA_NODES_NODE. This option can be used to connect an airbag model or belt anchorage points in LS-DYNA to a steering wheel model in MADYMO, for example.

Input Example Basic Coupling

The following is part of a MADYMO input deck for Basic Coupling:

```
<COUPLING>
   <COUPLING_SURFACE
      EXTERNAL_REF="1"
      SURFACE="/2/1"
      EXTERNAL_DATA="0.0"
   />
</COUPLING>
```

This will send data from surface /2/1 to LS-DYNA. In LS-DYNA one may refer to this data using reference 1. Note the arbitrary value 0.0 for EXTERNAL_DATA, which is not used in LS-DYNA.

The following is part of a LS-DYNA input deck for Basic Coupling:

```
*MAT_RIGID
         2 7.82000E+09 2.06800E+05 0.3000000         1         2         1         1
         0
*SECTION_SHELL
         1           0         0.0       0.0       0.0       0.0
       1.0         1.0         1.0       1.0       0.0
*PART
$  Part-ID  Section-ID Material-ID
      9001           1           2
```

Verification

The user is advised to check the log file and the reprint file to verify if all entities are communicated correctly. The LS-DYNA log file shows the LS-DYNA output of all entities received from MADYMO, the reprint file shows the entities MADYMO sends to LS-DYNA.

The LS-DYNA log file:

```
     MADYMO3D external number           = 1.00000E+00
     MADYMO3D coupling flag............ = 2.00000E+00
```

More information below these lines is printed in the LS-DYNA log file concerning the FE mesh builds on top of the ellipsoid. All of this information should be checked.

The accompanying MADYMO reprint file:

```
 COUPLING
    FE_MODEL ..................................:
      SURFACE .................................: /2/1
    EXTERNAL REFERENCE ........................:          1
    EXTERNAL DATA .............................:
    NUMBER ..:          1  DATA ..:      0.0000E+00
```

5.3.2 Extended Coupling

The coupling with LS-DYNA supports triangular and quadrilateral shell elements (*ELEMENT_SHELL), hexagonal solid elements (*ELEMENT_SOLID) and thick shell elements (*ELEMENT_TSHELL).

- Add the LS-DYNA *SET (*SET_PART, *SET_SHELL, *SET_SOLID) cards to define the LS-DYNA element sets to be used in the coupling contact interaction.
- Add the *CONTACT_COUPLING card to define the LS-DYNA sets (just defined) to be used for the contact in MADYMO. The format of *CONTACT_COUPLING is:

```
*CONTACT_COUPLING
   ID
   SID
   STYPE
```

where ID is the contact identifier; SID is the identifier of the SET to be used for coupling and STYPE is the SET type, which can have the following values:

0 : part set (default)
1 : shell element set
2 : solid element set
3 : thick shell set

Input Example Extended Coupling

The following is part of a MADYMO input deck.

```
<COUPLING
   FE_MODEL="/10/3"
/>
```

The following is part of a LS-DYNA input deck.

```
*CONTACT_COUPLING
      98
      63
*SET_PART
      63
      92 93
```

This forces LS-DYNA to send all elements belonging to part 92 and 93 to MADYMO.

Verification

The user is advised to check the log file and the reprint file to verify if all entities are communicated correctly. The log file shows the LS-DYNA output of all FE data send to MADYMO, the reprint file shows the FE data that MADYMO received from LS-DYNA.

Example LS-DYNA log file:

```
 c o n t a c t   c o u p l i n g

 contact coupling interface ID     =      98
 number of part IDs                =       1
 number of shell element IDs       =       0
 number of solid element IDs       =       0
 number of solid-shell element IDs =       0
```

```
shell element ID list for coupling:
        1         2         3         4

input summary including part IDs:
number of part IDs                =       9
number of unique nodal points     =       1
number of shell elements          =       4
number of solid elements          =       0
number of solid-shell elements    =       0
```

Accompanying reprint file:

```
COUPLING
   FE_MODEL ..............................: /10/3 ( /coupling_model/fe_model )
NUMBER OF NODES                          :          9
NUMBER OF PARTS                          :          1
  PART ID (FROM COUPLED PROGRAM)         :         92
    THICKNESS                            :      1.0000E-01
    BULK MODULUS                         :      2.5000E+07
NUMBER OF ELEMENTS                       :          4
  NUMBER OF TRIADS                       :          0
  NUMBER OF QUADS                        :          4
  NUMBER OF SOLIDS                       :          0
```

5.4 Coupling on Microsoft Windows platform

The new MPP coupling is also available on Microsoft Windows for LS-DYNA. The procedure to start a coupling calculation is:

1. Set the environment variables MADLIC_LICENSE_FILE and environment variables necessary to execute the LS-DYNA software.
2. Open two command shells and go to the directory where the input files for MADYMO and LS-DYNA are located. All input files must reside in the same directory, otherwise the communication between both programs will fail.
3. start MADYMO in command window 1 with the command:

   ```
   "<madymo_dir>\<platform_id>\bin\madymo_cli.exe" -coupling socket
   <input_xml_file>
   ```

4. start LS-DYNA in command window 2 with the command:

   ```
   "<lsdyna_dir>\lsdyna.exe" y=madymo i=<input_file>
   ```

5.5 Restart

For more general information on restart, see the LS-DYNA user manual and the MADYMO Reference Manual. Restart is supported in MPP coupling for both LS-DYNA and MPP-DYNA

version 5.0 and higher.

To enable restart, the following steps must be taken[1]

1. Generate restart files for LS-DYNA and MADYMO by running a normal, non-restart job.
2. Optionally: generate a LS-DYNA restart input deck (see the LS-DYNA manual for more information).
3. Generate a MADYMO restart input file (xml file). There is a template restart file in `<madymo_dir>/etc/etc/template_restart.xml`
4. Submit the restart job with the `-rst <restart.rst>` command line option for MADYMO, and the `r=<D3DUMPxxx>` command line option for LS-DYNA, where <restart.rst> points to the MADYMO restart file and <D3DUMPxxx> to the dyna restart file, both generated in step 1. For an example restart startup script, see section 5.5

5.5.1 Restart Control

- All remarks for normal coupling to LS-DYNA are valid for restart (see section 5.2).
- Both MADYMO and LS-DYNA restart file are cumulative.
- The restart frequency is controlled independently on both sides, but not communicated. The user must make sure that the frequencies are equal. The frequencies are controlled by the TIME_RESTART keyword in the MADYMO input deck and the *DATABASE_BINARY_D3DUMP keyword in the LS-DYNA input deck.
- Please check the log file that both LS-DYNA and MADYMO restart at the same timepoint. Selecting the wrong LS-DYNA and/or MADYMO restart file (see above) can result in out-of-sync restart runs. The simulation aborts when it encounters an out-of-sync simulation (and reports so).
- The restart analysis starts at the first timepoint saved in the restart file that is smaller than the begin time of the restart analysis.

5.6 Limitations and Known Issues for indirect coupling

- When the MADYMO end time is higher than the LS-DYNA end time, MADYMO can write the terminate line twice to the log file, with the first line:

  MADYMO TERMINATED NORMALLY.

  and the second line:

  MADYMO TERMINATED ABNORMALLY, because of ERRORS

[1] The user may read MPP-DYNA where LS-DYNA is mentioned.

The second line should not be written to the screen and log file, and is misleading.

In the reprint file, there is only one termination line and it should read:

MADYMO TERMINATED NORMALLY.

Workaround: set the two end times identical to each other.

- When the initial time step of LS-DYNA is larger than that of MADYMO, the simulation can get out of sync.

  Workaround: specify a smaller initial timestep than the MADYMO timestep, by changing the value of DTINT under *CONTROL_TIMESTEP in the LS-DYNA input file.

6 Coupling with PAM CRASH/SAFE

6.1 Introduction

This section gives specific information related to coupling with PAM CRASH/SAFE.

Please note that the details of the input deck for PAM CRASH/SAFE have been based on PAM CRASH/SAFE 2005 and may be subject to modifications. Always refer to the latest PAM CRASH/SAFE documentation for the correct syntax.

6.2 Simulation Control

- For general simulation control, please refer to Sec. 1.1.3.
- PAM CRASH/SAFE reads the number of CPU's it uses from the environment variable OMP_NUM_THREADS. This number must match the number of CPU's used by MADYMO, otherwise the simulation will terminate.

6.3 PAM CRASH/SAFE Input Deck Setup

This section describes the required modifications to the PAM CRASH/SAFE input deck to perform Basic Coupling or Extended Coupling simulations. See the PAM CRASH/SAFE User Manual for details on the cards not mentioned here specifically. Modifications to the MADYMO input deck are described in Sec. 3.

6.3.1 Basic Coupling

For Coupling either a coupling body or a coupling surface can be defined.

Body Coupling

- The value defined in EXTERNAL_DATA refers to the node that will be tied with the body center of gravity. In PAM CRASH/SAFE, this node can be defined as free or connectivity node. Post-processing of this node can be done in PAM CRASH/SAFE.

Surface Coupling

- EXTERNAL_DATA contains 2 values, i.e. EXTERNAL_DATA="5 3". The first value defined in EXTERNAL_DATA is used in PAM CRASH/SAFE to refer to part id's. Only in the ellipsoid case, the second value corresponds to the discretization density.

Define a PART and MATERIAL (type 100) section for all surfaces to be received from MADYMO. The PART identifiers should match to the first value of the accompanying EXTERNAL_DATA value in the MADYMO input deck. Note that PART and MATERIAL cards can be re-used. For further details, please refer to the PART SECTION and MATERIALS SECTION in the PAM CRASH/SAFE manual.

Input Example Basic Coupling

The following two examples are parts of a MADYMO input deck for Basic Coupling.

Body Coupling Example

```
<COUPLING>
   <COUPLING_BODY
      BODY = "/Vehicle/HUB"
      EXTERNAL_REF = "10"
      EXTERNAL_DATA = "30729"/>
</COUPLING>
```

The following is part of a PAM CRASH/SAFE input deck for Basic Coupling:

```
NODE  /    30729       2.7479956       0.3514169      0.82089777       0
SHELL /    30023     143   30720   30721   30729   30728       3 0.0
SHELL /    30024     143   30721   30722   30730   30729       3 0.0
SHELL /    30031     143   30728   30729   30737   30736       3 0.0
SHELL /    30032     143   30729   30730   30738   30737       3 0.0


$
```

Surface Coupling Example

```
<COUPLING>
   <COUPLING_SURFACE
      EXTERNAL_REF = "10"
      SURFACE = "/1/1"
      EXTERNAL_DATA = "1.0 3.0"/>
</COUPLING>
```

EXTERNAL_REF is ignored, although it is mandatory.

This will send data from surface /1/1 to PAM CRASH/SAFE. In PAM CRASH/SAFE one may refer to this data using reference 1. The density of the mesh created for this surface is 3.

The following is part of a PAM CRASH/SAFE input deck for Basic Coupling:

```
$
PART  /        1SHELL          1
NAME ELASTIC MEMBRANE
       0.0

 3.5000e-4
```

```
END_PART
MATER /        1     100           1000.       0       0       1       0
               0       0       0       0       0       0      1.       0
NAME ELASTIC MEMBRANE
  1000000.               0.35




$
```

Verification

The user is advised to check the log file and the reprint file to verify if all entities are communicated correctly. The log file shows the PAM CRASH/SAFE output of all entities received from MADYMO, the reprint file shows the entities MADYMO sends to PAM CRASH/SAFE.

The log file:

```
       *** SETUP MODE SUCCESSFULLY READ FROM MADYMO **


      NUMBER OF COUPLED ENTITIES .........         1
      NUMBER OF COUPLED ELLIPSOIDS .......         1
      NUMBER OF TIED NODES  ..............         0
      NUMBER OF PLANES ...................         0

      COUPLING ENTITY NO .................         1 (ELLIPSOID)
      SYSTEM NUMBER ......................         1
      ELLIPSOID NUMBER ...................         1
      ASSIGNED MATERIAL NUMBER (TYPE 100).         1
      AUTOMATIC MESH GENERATION DENSITY ..         3
      NUMBER OF GENERATED NODES ..........        92
      NUMBER OF GENERATED NULL SHELLS ....        90

      TOTAL NUMBER OF GENERATED NODES.....        92
      TOTAL NUMBER OF GENERATED SHELLS....        90
```

The accompanying MADYMO reprint file:

```
---------------------------------- COUPLING ----------------------------------

 COUPLING
    FE_MODEL ....................................:
      SURFACE ...................................: /2/1
    EXTERNAL REFERENCE ..........................:         10
    EXTERNAL DATA ...............................:
    NUMBER ..:          1  DATA ..:      1.0000E+00
    NUMBER ..:          2  DATA ..:      3.0000E+00

------------------------------- END COUPLING ---------------------------------
```

6.3.2 Extended Coupling

- Define the MDBODY cards; please refer to the PAM CRASH/SAFE Solver Reference Manual for detailed information. Briefly, the MDBODY card contains the following entities

```
$---5---10----5---20----5---30----5---40----5---50----5---60----5---70----5---80
$       IDMDBO  CTYPE   BULK    THICK   IDCVS--|
MDBODY/
$   TITLE
NAME
```

where the relevant control parameters for the coupling with MADYMO are

- IDMDBO, i.e. the FE deformable body identification number
- CTYPE, i.e. the FE mesh type (SHELL, SOLID or NODE)
- BULK, i.e. the average bulk modulus to be used by MADYMO.
- THICK, i.e. the average thickness to be used by MADYMO.
- IDCVS, i.e. external identifier to be referenced in MADYMO.
- Every MDBODY card is followed by cards to define element or node selection by keywords (See General Entity Selection sub-section in the CONTROL SECTION of the PAM CRASH/SAFE manual)

- The "COUPLING MADYMO" keyword is mandatory in the control section.
- This card should be repeated for each FE Deformable Body to be sent to MADYMO.
- If the average bulk modulus and/or the average thickness are not defined, average values will be computed by PAM CRASH/SAFE.
- The identifiers IDCVS, set in the MDBODY cards, can be referred to in the MADYMO input deck as part id's.
- If MDBODY type equals NODE, the IDCVS label can not be used in MADYMO. Instead, one must refer to the individual node numbers. Note that element and part information related to the communicated nodes is not send to MADYMO.

Input Example Extended Coupling

The following is part of a MADYMO input deck.

```
<COUPLING
   FE_MODEL="/Coupling_System/External_Boule"
/>
```

The following is part of a PAM CRASH/SAFE input deck.

```
$---5---10----5---20----5---30----5---40----5---50----5---60----5---70----5---80
$       IDMDBO  CTYPE   BULK    THICK   IDCVS--|
MDBODY/        2SHELL                          3
NAME boule
        PART 1
        END
```

This forces PAM CRASH/SAFE to send all shell elements belonging to part 1 to MADYMO. Average bulk modulus and thickness are calculated by PAM CRASH/SAFE. In MADYMO the elements can be referred to either by element number or by the reference number 3(!).

Verification

The user is advised to check the log file and the reprint file to verify if all entities are communicated correctly. The log file shows the PAM CRASH/SAFE output of all FE data send to MADYMO, the reprint file shows the FE data that MADYMO received from PAM CRASH/SAFE.

Example log file:

```
 F E   B O D Y   F O R   M A D Y M O


     FE DEFORMABLE BODY ID ............ =           2
     FE DEFORMABLE BODY NAME .......... = boule
     BODY MESH TYPE ................... = SHELL
     REFERENCE ID IN MADYMO CONTACT ... =           3
     AVERAGE BULK MODULUS ............. =  0.1111E+10
     AVERAGE THICKNESS ................ =  0.3500E-03


     NUMBER OF ASSOCIATED SHELLS ...... =          90

               LIST OF SHELLS TRANSFERED TO MADYMO:


 ====  DATA SUPPRESSED BY NOPRINT OPTION  ====
```

Accompanying reprint file:

```
----------------------------COUPLING INTERFACE------------------------------
 NUMBER OF NODES                       :          92
 NUMBER OF PARTS                       :           1
   PART ID (FROM COUPLED PROGRAM)      :           3
     THICKNESS                         :      3.5000E-04
     BULK MODULUS                      :      1.1111E+09
 NUMBER OF ELEMENTS                    :          90
   NUMBER OF TRIADS                    :           0
   NUMBER OF QUADS                     :          90
   NUMBER OF SOLIDS                    :           0
```

6.4 Limitations and Known Issues

- The last DSY file might not be written if it occurs at the MADYMO simulation end time point (you can slightly increase the MADYMO simulation end time to avoid this problem).

7 Coupling with Radioss

7.1 Introduction

This section gives specific information related to the coupling with Radioss.

Please note that the details of the input deck for Radioss have been based on Radioss 4.7.4 and may be subject to modifications. Always refer to the latest Radioss documentation for the correct syntax.

7.2 Simulation Control

- For general simulation control, please refer to Sec. 1.1.3.
- For Direct coupling, the simulation end time point defined in the MADYMO input file is considered only. The end time point defined in the Radioss file is ignored.

  For MPP coupling, the simulation ends when either the simulation end time point specified in the MADYMO input file or the Radioss input file is reached, whichever is lower.
- The /MODIF Radioss's option is not suppported while using coupling.
- To enable direct coupling, the option /MADYMO/ON in the Radioss ENGINE input file is necessary.

  To enable MPP coupling, the option /MADYMO/MPP in the Radioss ENGINE input file is necessary.

```
/MADYMO/ON
Lunit Tunit Munit
```

Lunit	Length unit conversion factor, must be 1 meter in Radioss model units
Tunit	Time unit conversion factor, must be 1 s in Radioss model units
Munit	Mass unit conversion factor, must be 1 Kg in Radioss model units

```
/MADYMO/ON
1   1000  1
```

Here the 1000 specifies that Radioss should divide their time values with 1000 before sending them to MADYMO (and the inverse for all time numbers received from MADYMO), effectively stating that the Radioss input file is in ms. The lenght and force numbers are unchanged.

7.3 Radioss Input Deck Setup

This section describes the required modifications to the Radioss input deck to perform Extended Coupling or Basic Coupling simulations. See the Radioss User Manual for details on the cards not mentioned here specifically. Modifications to the MADYMO input deck are described in Sec. 3.

7.3.1 Basic Coupling

- Note that Radioss only uses the value of EXTERNAL_REF. However, the attribute EXTERNAL_DATA is mandatory in MADYMO and must therefore be given a value, be it arbitrary.
- For Radioss, planes will be output in Radioss ANIMATION files, but no contact computation is possible with Radioss FE in Radioss.
- The interfaces TYPE 14 and 15 allow to compute contact between MADYMO hyperellipsoids and Radioss FE. The option /MADYMO/LINK allows to connect a Radioss node to a MADYMO body. (See the Radioss CRASH input manual for a description of the options /MADYMO/LINK, /SURF/MDELLIPS, /INTER/TYPE14 and /INTER/-TYPE15).

Input Example Basic Coupling

The following is part of a MADYMO input deck for Basic Coupling:

```
<COUPLING>
   <COUPLING_SURFACE
      EXTERNAL_REF="10"
      SURFACE="/2/1"
      EXTERNAL_DATA="0.0"/>
</COUPLING>
```

This will send data from surface /2/1 to Radioss. In Radioss one may refer to this data using reference 10. Note the arbitrary value 0.0 for EXTERNAL_DATA, which is not used in Radioss.

The following is part of a Radioss STARTER input deck for Basic Coupling:

```
/SURF/MDELLIPS/1001/The first madymo coupl'd surface
      10
```

Verification

The user is advised to check the log file and the reprint file to verify if all entities are communicated correctly. The log file shows the Radioss output of all entities received from MADYMO, the reprint file shows the entities MADYMO sends to Radioss.

The Radioss STARTER log file:

```
 EXTERNAL COUPLING TO SURFACE:    1001 The first madymo coupl'd surface
 ---------------------------

          SURFACE DEFINITION :
          --------------------
          REFERENCE TO MADYMO COUPLING SURFACE :       10
          CORRESPONDING DATA WILL BE READ IN RADIOSS ENGINE.
```

The accompanying MADYMO reprint file:

```
---------------------------------- COUPLING ----------------------------------
 COUPLING
    FE_MODEL ........................:
      SURFACE .......................: /2/1 ( /S2_first_ball/ell1_first_ball )
    EXTERNAL REFERENCE ..............:         10
    EXTERNAL DATA ...................:
    NUMBER ..:          1  DATA ..:      1.0000E+03
---------------------------------- END COUPLING ------------------------------
```

7.3.2 Extended Coupling

- MADYMO and Radioss extended coupling is available with Block Format 4.4 for Radioss STARTER with additional functionalities.
- The following additional option is available in Radioss version 4.7:

  ```
  /MADYMO/EXFEM/exfem_id/exfem_title
  ```

  in which exfem_id is the FEM identification number and exfem_title is the FEM name. Following this option, one has to specify the part id's of the elements of which information is exchanged with MADYMO:

  1 to n CARDS:

  ```
  1         2          3       ...
  Ip1       Ip2        Ip3     ...
  ```

  in which Ip1, Ip2, etc are the integer part id's.

  - exfem_id and exfem_title are not exchanged with MADYMO.
  - Part IDs, as well as related nodes IDs and elements IDs can be used in the MADYMO input file, in order to define contact interfaces between these Radioss entities and MADYMO MB or FE models. These parts must be parts of shells, 3 nodes shells or 8 nodes bricks.
  - Options /MADYMO/LINK cannot use any node belonging to these parts. Radioss STARTER will generate an error in such a case.
  - A node belonging to such a part can not be a slave node of an interface TYPE 2, nor of a rigid body within Radioss, if it receives contact forces from MADYMO contact interfaces: Radioss STARTER writes a warning if a node belonging to the exchanged parts is a slave node of an interface TYPE 2 or of a rigid body.

Input Example Extended Coupling

The following is part of a MADYMO input deck.

```
<COUPLING
   FE_MODEL="/1/100"
>
```

The following is part of a Radioss input deck.

```
/MADYMO/EXFEM/1/
       2
```

This forces Radioss to send all information of nodes and elements belonging to part 2 to MADYMO.

Verification

The user is advised to check the log file and the reprint file to verify if all entities are communicated correctly. The Radioss log file shows the Radioss output of all FE data send to MADYMO, the reprint file shows the FE data that MADYMO received from Radioss.

Example Radioss log file:

```
      FEM INTERFACED TO MADYMO DEFINITION
      -----------------------------------


      FOLLOWING PARTS         WILL BE SENT TO MADYMO
       2

      FOLLOWING 4-NODES SHELL WILL BE SENT TO MADYMO
    4427    4428    4429    4430

      FOLLOWING 3-NODES SHELL WILL BE SENT TO MADYMO


      FOLLOWING BRICKS        WILL BE SENT TO MADYMO


      FOLLOWING NODES         WILL BE SENT TO MADYMO
    4308    4309    4310    4311    4312    4313    4314    4315    4316
```

Accompanying reprint file:

```
----------------------------COUPLING INTERFACE------------------------------
 NUMBER OF NODES                       :          9
 NUMBER OF PARTS                       :          1
   PART ID (FROM COUPLED PROGRAM)      :          2
     THICKNESS                         :      8.1000E-04
     BULK MODULUS                      :      6.9608E+10
 NUMBER OF ELEMENTS                    :          4
   NUMBER OF TRIADS                    :          0
   NUMBER OF QUADS                     :          4
   NUMBER OF SOLIDS                    :          0
```

7.4 Restart

For more general information on restart, see the Radioss engine manual and the MADYMO Reference Manual. Restart is supported for direct coupling only.

To enable restart, the following steps must be taken:

1. Generate restart files for Radioss and MADYMO.
2. Generate a Radioss engine input deck (D02 file). In the D02 file, set the identifier to `/RUN/ROOTNAME/<restart_id>`. Change the <restart_id> to select the desired Radioss restart file. Radioss restart files are cyclic. The number of Radioss restart files is determined by the `/RFILE/n` option in the engine file, where `n` determines the number of restart files.
3. Generate a MADYMO restart input file (xml file). There is a template restart file in `<madymo_dir>/etc/etc/template_restart.xml`.
4. Submit the restart job with the `-rst <restart.rst>` command line option, where <restart.rst> points to the restart file generated in step 1. For an example restart startup script, see section 8.1.4

7.4.1 Restart Control

- All remarks for normal coupling to Radioss are valid for restart (see section 7.2).
- The MADYMO restart file is cumulative. All time points for Radioss are written in one file.
- The restart frequency is fully MADYMO controlled, and is enabled by the TIME_RESTART keyword in the MADYMO input deck. The TIME_RESTART and D01 RFILE don't need to be synchronized. MADYMO will write a restart file every TIME_RESTART seconds, and Radioss follows this.
- Please check the log file that both Radioss and MADYMO restart at the same time-point. Selecting the wrong Radioss restart file (see above) can result in out-of-sync restart runs.
- The restart analysis starts at the first timepoint saved in the restart file that is smaller than the begin time of the restart analysis.

7.5 Limitations and Known Issues

- The last ANIMATION file may not be written if the simulation end time is a multiple of the output time step. This situation can be avoided by increasing the simulation end time point slightly.

- Running a coupling executable between Radioss and MADYMO is not possible through the RADTOOL.

8 Example Startup Scripts

This section shows some examples to run coupling executables with LS-DYNA, MPP-DYNA, PAM CRASH/SAFE and Radioss.

8.1 Direct coupling

In general, the direct coupling is started with the following command line:

```
madymocli -coupling direct <input.xml> -x /path/to/coupling_exe -arg <partner args>
```

where <input.xml> is the MADYMO input deck, /path/to/coupling_exe is the PARTNER coupling executable and <partner args> are all arguments needed by this coupling executable.

The benefit of this approach is that MADYMO handles all environment variables that MADYMO needs to run in coupling, including adapting the run time library search path ($LIBPATH, $LD_LIBRARY_PATH or $SHLIB_PATH, depending on the OS).

8.1.1 PAM CRASH/SAFE

```
#!/bin/sh
# example script to run MADYMO/PAM coupling
# use with options: input.xml input.pc #cpu
#
PAMEXE=/path/to/pam2006/psolid.x
# PAMLIB is usually a subdir of the dir where psolid.x resides
PAMLIB=/path/to/pam2006/lib
XML=$1
PC=$2
CPU=$3
export OMP_NUM_THREADS=${CPU}
export PAMLIC=CRASHSAF
export LD_LIBRARY_PATH=${PAMLIB}
# assuming madymo75 is in the search path
madymo75 -coupling direct -nrproc ${CPU} -x ${PAMEXE} ${XML} -arg ${PC}
```

8.1.2 LS-DYNA

```
#!/bin/sh
# example script to run MADYMO/LSDYNA coupling
# use with options: <input.xml> <input.key>
XML=$1
K=$2
# lsdyna executable
LSEXE=/path/to/ls970.madymo75
# assuming madymo75 is in the search path
madymo75 -coupling direct $XML -x ${LSEXE} -arg y=madymo i="$K"
```

8.1.3 MPP-DYNA

MPP-DYNA with lammpi

```
#!/bin/sh
# example script to run MADYMO/MPPDYNA coupling
# use with options: <input.xml> <input.key> <#nodes for mppdyna>
XML=$1
K=$2
NP=$3
MPPEXE=/path/to/mpp971.madymo75
# make sure lammpi is in the path
PATH=/usr/local/mpi/lam-6.5.9/bin:${PATH};  export PATH
#
# build wrapper script for lam
# so that MADYMO env.vars are known at boot-time of the the lam universe
cat > ./lamrun.$$ << _EOF_
#!/bin/sh
lamboot -v
mpirun -np $NP $MPPEXE y=madymo i=$K $XML
wipe -v
_EOF_
#
chmod 755 ./lamrun.$$
# assuming madymo75 is in the search path
madymo75 -nopretrans -coupling direct $XML -x ./lamrun.$$
# cleanup
rm -f ./lamrun.$$
```

MPP-DYNA with mpich/native mpi

```
#!/bin/sh
# example script to run MADYMO/MPPDYNA coupling
# use with options: <input.xml> <input.key> <#nodes for mppdyna>
XML=$1
K=$2
NP=$3
# mppdyna executable
MPPEXE=/path/to/mpp971.madymo75
MPIRUN=/opt/mpi/bin/mpirun
# assuming madymo75 is in the search path
madymo75 -coupling direct $XML -x $MPIRUN -arg -np $NP $MPPEXE i=$K y=madymo
```

MPP-DYNA with mpi that doens’t propagate env. vars

```
#!/bin/sh
# example script to run MADYMO/MPPDYNA coupling
# use with options: <input.xml> <input.key> <#nodes for mppdyna>
XML=$1
K=$2
NP=$3
MPPEXE=/path/to/mpp971.madymo75
# make sure lammpi is in the path
MPIRUN=/usr/local/mpi/bin/mpirun
#
# build wrapper script for mpi
# to force setting env.vars before starting executable
cat > ./mpirun.$$ << _EOF_
#!/bin/sh
# wrapper script for dumb mpi
export LD_LIBRARY_PATH=\$LD_LIBRARY_PATH
```

```
export MADETCPATH=\$MADETCPATH
export MADINCPATH=\$MADINCPATH
export MADBINPATH=\$MADBINPATH
mpirun -np $NP $MPPEXE y=madymo i=$K $XML
_EOF_
#
chmod 755 ./mpirun.$$
# assuming madymo75 is in the search path
madymo75 -nopretrans -coupling direct $XML -x ./mpirun.$$
# cleanup
rm -f ./mpirun.$$
```

MPP-DYNA with LSF on IBM

```
#!/bin/sh
# example script to run MADYMO/MPPDYNA coupling
# use with options: <input.xml> <input.key>
XML=$1
K=$2
# mppdyna executable
MPPEXE=/path/to/mpp971.madymo75
# start coupling through LSF pam
# assuming madymo75 is in the search path
madymo75 -coupling direct $XML -x pam -arg -g -1 poejob $MPPEXE i=$K y=madymo
```

8.1.4 Radioss

Radioss smp

```
#!/bin/sh
# example script to run MADYMO/Radioss coupling
# use with args: <input.xml> <input> (without extension!)
XML=$1
D00=$2.D00
D01=$2.D01
if ! [ -h file_engine ]
then
#   change this to make a softlink from the correct file_engine file
  ln -s  /path/to/staff/file_engine file_engine
fi
# set STARTER to Radioss starter
export STARTER=/path/to/radioss_exe/s474x_spmd
# set RADMAD to the coupled executable
export RADMAD=/path/to/rad2md/d474x_m64_pmd
#
${STARTER} < $D00
#
# assuming madymo75 is in the search path
madymo75  -coupling direct -x ${RADMAD} $1 < $D01
```

Radioss mpp

```
#!/bin/sh
# use with args: <input.xml> <input.D00> <input.D01> (including extension!)
XML=$1
D00=$2
D01=$3
if ! [ -h file_engine ]
```

```
then
#   change this to make a softlink from the correct file_engine file
  ln -s  /path/to/staff/file_engine file_engine
fi
# set STARTER to radioss starter
export STARTER=/path/to/radioss_exe/s474x_spmd
# set RADMAD to the coupled executable
export RADMAD=/path/to/rad2md/d474x_m64_pmd
#
${STARTER} < $D00
#
# assuming madymo75 is in the search path
madymo75 -coupling direct -x ${RADMAD} $XML -arg -p4pg hostsfile -p4wd ‘pwd‘ < $D01
```

Radioss mpp and restart

```
#!/bin/sh
# use with args: <input.xml> <restart.rst> <input.D02> (including extension!)
XML=$1
RST=$2
D02=$3
if ! [ -h file_engine ]
then
#   change this to make a softlink from the correct file_engine file
  ln -s  /path/to/staff/file_engine file_engine
fi
# starter not needed, allready done in normal run
# set RADMAD to the coupled executable
export RADMAD=/path/to/rad2md/d474x_m64_pmd

# assuming madymo75 is in the search path
# the next line must be on one line, but maybe textwrapping changes this
madymo75 -coupling direct -rst ${RST} -x ${RADMAD} $XML -arg -p4pg hostsfile -p4wd
‘pwd‘ < $D02
```

8.2 MPP coupling

8.2.1 Introduction

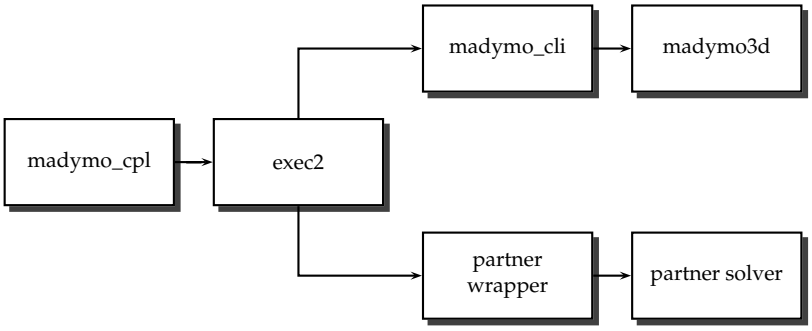
To enable users to easily start MADYMO and the PARTNER, the madymo_cpl program has been developed. The madymo_cpl program is able to start two MADCL-enabled solvers at the same time, in one shell on Linux platforms.

8.2.2 What does it do

The madymo_cpl program calls the `exec2` binary, which is shipped with the MADYMO distribution. Then `exec2` starts both the MADYMO and the PARTNER solver. In case one of the solvers exits with an error condition, the `exec2` program will terminate the other solver.

The process tree is given below:

The way the PARTNER solver is started differs per partner, i.e., madymo_cpl will use a stan-



dard parameter list and the user will have to write a wrapper script around the PARTNER solver. This script will have to parse the arguments supplied by madymo_cpl, and generate a valid command to actually start the PARTNER solver. The argument interface will be described in Sec. 8.2.6.

8.2.3 Usage

In general, the MPP coupling is started with the following command line where the madymo_cpl start-up script is used:

```
madymo_cpl [options] <madymo input file> <FE partner input file>
```

The options available for the madymo_cpl script are

-help	Print all options of madymo_cpl
-config <config_file>	Load settings from configuration file <config_file>
-febin <exe>	Specifies the FE executable.
-npmad <NPMAD>	Run madymo with <NPMAD> processes.
-npfe <NPFE>	Run the FE executable with <NPFE> processes.
-coupling <cpltype>	Selects the coupling communication transport. Valid values are `socket` and `mpi`. The default is `mpi`.
-mpilib <mpi-impl>	Selects the mpi implementation. See also the -mpilib option of madymo_cli. Depending on your machine architecture, valid values may be mpp and hpmpi. The default is hpmpi.
-hostfile <hostfile>	Host file for madymo_cpl. Please read madymo_cpl documentation on how to write a hostfile.
-madargs ... –	Specifies any additional madymo arguments, e.g., -isize, -rsize, and -csize.
-feargs ... –	Specifies any additional FE arguments. Valid arguments depend on the -febin option.

The configuration file format is described in Sec. 8.2.5. Either a configuration file is passed to madymo_cpl, or arguments are passed on the command line. The 'hostfile', 'madargs' and 'feargs' arguments are optional; the 'coupling' and 'mpilib' arguments have default values (respectively 'mpi' and 'hpmpi'). If 'coupling' is set to 'socket', the 'mpi' argument is ignored.

IMPORTANT: If the PARTNER solver makes use of the HP-MPI capabilities of MADCL, an environment variable `MPIRUN` has to be set prior to starting madymo_cpl. This variable should point to the 'mpirun' executable. In Sec. 8.2.6 it is explained how madymo_cpl handles the `MPIRUN` variabe.

8.2.4 Hostfiles

When HP-MPI is used to use more than one machine for the simulation (like on a cluster), HP-MPI must be told what machines/nodes it can use. The list of machines the solvers can run on has to be passed to madymo_cpl by means of a hostfile; this is a text file, similar to a standard HP-MPI hostfile. However, since madymo_cpl starts two solvers, the hostfile is split into two sections: one for MADYMO, and one for the PARTNER. Each section lists the hosts available to its respective solver. The whole hostfile is parsed by madymo_cpl, and split into two separate hostfiles (one for each solver). A hostfile should look like this:

```
[MADYMO]
```

Option	Corresponds to
FEBIN	-febin
NPMAD	-npmad
NPFE	-npfe
COUPLING	-coupling
MPILIB	-mpilib
HOSTFILE	-hostfile
MADARGS	-madargs
FEARGS	-feargs
MADINPUT	MADYMO input file
FEINPUT	PARTNER input file

```
host1:np1
host2:np2

[FEPARTNER]
host3:np3
host4:np4
```

The 'hostX' strings should be replaced the host names of the available machines, and the 'npX' strings should be replaced with the number of CPUs the solver can use on that host. The number of hosts per section is arbitrary, and the 'npX' values depend on the host capabilities.

Note:

1. The total number of spawned processes (being <npmad> + <npfe>) should not exceed the sum of all 'npX' values. If more processes are spawned on a machine/node than there are available CPUs, either HP-MPI will not run at all, or there will be a severe performance penalty!
2. If no 'hostfile' option is passed to madymo_cpl, the MADYMO and PARTNER solvers will be started on the machine/node where madymo_cpl was started (local run). Again, please take into consideration how many CPUs are available on the machine! Of course a hostfile can be written with just one host per solver, to 'offload' the simulation to another (single) host.

8.2.5 Configuration Files

A configuration file is a simple text file, with one option per line. An option is formatted as "OPTION=VALUE". Valid options are shown in the table below.

There is no need to put all options in a configuration file, i.e., a few options can be set in the configuration file and the rest of the options can be set on the command line. If an option is defined in both the configuration file as well as on the command line, the command line value overrules the value in the configuration file.

8.2.6 Argument Interface

When madymo_cpl calls exec2 to start both solvers, a standard argument list is passed to the PARTNER solver. Of course not all PARTNER solvers are started in the same way. This means a wrapper script will have to be written, as intermediate layer; the 'febin' option should be set to point to this wrapper script. The wrapper script is started from exec2 with this command:

```
<febin> -nrproc <npfe> -input <feinput> <feargs>
```

The wrapper script will have to parse these arguments into a valid command to actually start the PARTNER solver and/or `MPIRUN`. If HP-MPI is used, the PARTNER solver wrapper script is started in an environment where a variable FEMPIRUN is set; FEMPIRUN points to the 'mpirun' executable, with a number of options:

```
FEMPIRUN = "<mpirun> -spawn -prot -nameserver <nameserver> [-hostfile <PARTNER hostfile>]"
```

The wrapper script then has to extend the FEMPIRUN variable with exports for the PARTNER solver (like license variables, LD_LIBRARY_PATH, etc), other desired HP- MPI arguments, the PARTNER solver executable, and the 'feargs' string.

Example wrapper scripts for LS-Dyna, MPP-Dyna, PAMCrash, and RADIOSS are given in Sec. 8.2.9.

8.2.7 Checklist and Example

These are the steps to start a MADCL coupled run:

- Write or obtain a wrapper script for the PARTNER solver
- When using HP-MPI, set the MPIRUN environment variable
- When using HP-MPI and multiple hosts are available, write a host file
- If needed/desired, write a configuration file

As an example, a madymo_cpl command could look like this:

```
<madymo bin dir>/madymo_cpl -febin <script dir>/cpl_partner.sh \
    -npmad 4 -npfe 28 -coupling mpi -mpilib hpmpi              \
    -hostfile myhostfile -madargs <madymo args> --             \
    -feargs <partner args> -- <xml deck> <partner deck>
```

Now madymo_cpl will:

- Use MADYMO executable in "<madymo bin dir>"
- Use "<script dir>/cpl_partner.sh" as PARTNER wrapper script

- Start 4 MADYMO processes
- Start 28 PARTNER processes
- Use MPI (instead of old, direct coupling), and more specifically: use MADCL for HP-MPI
- Use the hosts listed in "myhostfile"
- Pass "<madymo args>" to MADYMO as extra arguments
- Pass "<partner args>" to PARTNER as extra arguments
- Use "<xml deck>" and "<partner deck>" as input files.

8.2.8 Restart

To perform restart jobs, first launch partner programs as usual. Next, launch a second job with restart command line options. As an example, a madymo_cpl command to run a restart simulation with LS-DYNA could look like this:

```
madymo_cpl -febin lsdyna_cpl.sh -coupling socket input.xml input.k
madymo_cpl -febin lsdyna_cpl.sh -coupling socket \
    -madargs -rst input.rst -- -feargs r=d3dump01 -- input.restart.xml input.restart.k
```

Now madymo_cpl will first:

- Use lsdyna_cpl.sh as LS-DYNA wrapper script.
- Use socket version of MADCL.
- Use input.xml and input.k as input files.

and then in the next run:

- Use lsdyna_cpl.sh as LS-DYNA wrapper script.
- Use socket version of MADCL.
- Pass "-rst input.rst" to MADYMO as extra arguments
- Pass "r=d3dump01" to LS-DYNA as extra arguments
- Use input.restart.xml and input.restart.k as input files.

8.2.9 Example wrapper scripts

This section gives example wrapper scripts for the PARTNER codes. These scripts are also available in the MADYMO distribution.

LS-Dyna

```
#! /bin/sh

# (c) TASS, 2010
#
# This script should be called from madymo_cpl. It is not meant to
# be used standalone. Use this script as FE partner executable for
# madymo_cpl.
#
# This script will launch LS-Dyna as FE partner for MADYMO.

# Initialize variables; change them if needed
MACHARCH=${MACHARCH:=linux26-x86_64}
DYNAEXE=${DYNAEXE:=/mnt/usr6/people/madymo/mppcoupling/lsdyna/${MACHARCH}/lsdyna/lsdyna_971}
export LSTC_FILE=${LSTC_FILE:=network}
export LSTC_LICENSE_SERVER=${LSTC_LICENSE_SERVER:=vxlic}
export LSTC_MEMORY=auto

### DON'T CHANGE ANYTHING BELOW THIS LINE ###

# Default number of processes
ncpu="ncpu=1"

# Parse the command line arguments.
job_args=""
job_name=""
while [ $# -gt 0 ]; do
    case "$1" in
        (-nrproc)
            shift
            ncpu="ncpu=$1" ;;
        (-input)
            shift
            job_name="$1" ;;
        (*)
            job_args="$job_args $1" ;;
    esac
    shift
done

# Construct command, echo it, and run
cmd="$DYNAEXE y=madymo $ncpu i=$job_name $job_args"
echo "Running command: $cmd"
eval $cmd
```

MPP-Dyna

```
#! /bin/sh

# (c) TASS, 2010
#
# This script should be called from madymo_cpl. It is not meant to
# be used standalone. Use this script as FE partner executable for
# madymo_cpl.
#
# This script will launch MPP-Dyna as FE partner for MADYMO.

# Initialize variables; change them if needed
MACHARCH=${MACHARCH:=linux26-x86_64}
DYNAEXE=${DYNAEXE:=/mnt/usr6/people/madymo/mppcoupling/mppdyna/${MACHARCH}/mppdyna/mppdyna971}
LSTC_FILE=${LSTC_FILE:=network}
LSTC_LICENSE_SERVER=${LSTC_LICENSE_SERVER:=vxlic}
LSTC_MEMORY=auto
```

```
# LD_LIBRARY_PATH has to include libmadcl.so
LD_LIBRARY_PATH=/usr1/people/madymo/linux26-x86_64/ext/lib-hpmpi:${LD_LIBRRAY_PATH}

### DON'T CHANGE ANYTHING BELOW THIS LINE ###

np="-np 1"
ncpu="ncpu=1"

# Parse the command line arguments.
job_name=""
job_args=""
while [ $# -gt 0 ]; do
    case "$1" in
        (-nrproc)
            shift
            np="-np $1"
            ncpu="ncpu=$1" ;;
        (-input)
            shift
            job_name="$1" ;;
        (*)
            job_args="$job_args $1" ;;
    esac
    shift
done

# Construct command, echo it, and run
cmd="$FEMPIRUN $np -e MPI_WORKDIR=‘pwd‘ \
       -e LSTC_FILE=$LSTC_FILE -e LSTC_LICENSE_SERVER=$LSTC_LICENSE_SERVER \
       -e LD_LIBRARY_PATH=$LD_LIBRARY_PATH -e LSTC_MEMORY=${LSTC_MEMORY}\
       $DYNAEXE $ncpu y=madymo i=$job_name $job_args"
echo "Running command: $cmd"
eval $cmd
```

Radioss

```
#! /bin/sh

# (c) TASS, 2010
#
# This script should be called from madymo_cpl. It is not meant to
# be used standalone. Use this script as FE partner executable for
# madymo_cpl.
#
# This script will launch RADIOSS as FE partner for MADYMO. This
# script expects the *D00 / *D01 filename format; leave out the 00/01
# part of the filename when starting madymo_cpl. So, for example, if
# you have two files testD00 and testD01, use testD as input filename.
#
# This script can skip the RADIOSS starter. To do so, use "-nostarter"
# as FE partner argument for madymo_cpl:
#     <path_to_mad_bin>/madymo_cpl [..] -feargs -nostarter [..] -- [..]

# Initialize variables; change them if needed
MACHARCH=${MACHARCH:=linux26-x86_64}
RADDIR=/mnt/usr6/people/madymo/mppcoupling/radioss/${MACHARCH}/radioss
RADFLEXFILE=radflex10_linux
RADFLEXSRC=$RADDIR/$RADFLEXFILE
RADSTARTER=$RADDIR/radioss_starter
RADSOLVER=$RADDIR/radioss_solver
export ALTAIR_LM_LICENSE_FILE="@sw013"
export ALTAIR_LICENSE_PATH=${ALTAIR_LICENSE_PATH:="6200@vxlic"}
export LD_LIBRARY_PATH=$RADDIR:$LD_LIBRARY_PATH
```

```
### DON'T CHANGE ANYTHING BELOW THIS LINE ###

# Default number of processes
np="-np 1"
# By default, run the starter too
use_starter=1

# Parse the command line arguments.
job_args=""
job_name=""
while [ $# -gt 0 ]; do
    case "$1" in
        (-nrproc)
            shift
            np="-np $1" ;;
        (-input)
            shift
            job_name="$1" ;;
        (-nostarter)
            use_starter=0 ;;
        (*)
            job_args="$job_args $1" ;;
    esac
    shift
done

# Check for flex file; if needed, symlink it
if [ ! -h $RADFLEXFILE ]; then
    ln -s $RADFLEXSRC .
fi

# Check whether the RADIOSS starter has to run
if [ "$use_starter" = "1" ]
then
    # Run starter
    cmd="$RADSTARTER -i ${job_name}00"
    echo "Running starter: $cmd"
    eval $cmd

    # Check whether the starter completed successfully
    if [ $?  -ne 0 ]
    then
        # In case of errors, exit. exec2 will terminate madymo
        echo "Error starting starter, aborting"
        exit 1
    fi
fi

# Construct command, echo it, and run
cmd="${FEMPIRUN} -e MPI_WORKDIR=‘pwd‘ \
       -e LD_LIBRARY_PATH=${LD_LIBRARY_PATH} \
       -e ALTAIR_LM_LICENSE_FILE=${ALTAIR_LM_LICENSE_FILE} \
       -e ALTAIR_LICENSE_PATH=${ALTAIR_LICENSE_PATH} \
       ${np} ${RADSOLVER} -i ${job_name}01 $job_args"
echo "Running command: $cmd"
eval $cmd
```

Pam Crash

```
#!/bin/sh

# (c) TASS, 2010
#
# This script should be called from madymo_cpl. It is not meant to
# be used standalone. Use this script as FE partner executable for
```

```
# madymo_cpl.
#
# This script will launch PAM-Crash as FE partner for MADYMO.

# Initialize variables; change them if needed
MACHARCH=${MACHARCH:=linux26-x86_64}
PAM_LMD_LICENSE_FILE=${PAM_LMD_LICENSE_FILE:=27022@vxlic}
export PAMLIC=${PAMLIC:=CRASHSAF}
PCDIR=/mnt/usr6/people/madymo/mppcoupling/pamcrash/${MACHARCH}/pamcrash
PCEXE=$PCDIR/psolid.x
PCLIB=$PCDIR/lib
# Include libmadcl.so and PAM-Crash libraries in LD_LIBRARY_PATH
LD_LIBRARY_PATH=/usr1/people/madymo/${MACHARCH}/ext/lib-hpmpi:$PCLIB:${LD_LIBRARY_PATH}

### DON'T CHANGE ANYTHING BELOW THIS LINE ###

# Default number of processes
NP="-np 1"

# Parse the command line arguments.
job_args=""
job_name=""
while [ $# -gt 0 ]; do
    case "$1" in
        (-nrproc)
            shift
            NP="-np $1" ;;
        (-input)
            shift
            job_name=$1 ;;
        (*)
            job_args="$job_args $1" ;;
    esac
    shift
done

# Construct command, echo it, and run
CMD="$FEMPIRUN -e LD_LIBRARY_PATH=$LD_LIBRARY_PATH -e MPI_WORKDIR=‘pwd‘ \
    -e PAM_LMD_LICENSE_FILE=$PAM_LMD_LICENSE_FILE -spawn -prot \
    -e PAMLIC=$PAMLIC $NP $PCEXE -coupled_MAD_MPI $job_args $job_name"
echo "Running command: $CMD"
eval $CMD
```

8.2.10 Troubleshooting

The MPP coupling may fail to start properly and report an error. A description of the errors typically encountered is given here.

1. undefined symbol: MPI_Lookup_name

   The user used a wrong version of HPMPI and/or is pointing LD_LIBRARY_PATH to the wrong libmpirun.so library.

2. MPI_Lookup_name: Dynamic Process is not turned on.
   MPI_Lookup_name: Unclassified error

   This is caused by wrong MPIRUN settings. The user

   - needs to point MPIRUN to the mpirun binary only (and possibly settings for the protocol). No settings for hostfiles and such shall be included, since this may clash

with settings that madymo_cpl adds later.

- FEMPIRUN needs NOT be set in the wrapper scripts for partners. FEMPIRUN is set my madymo_cpl and child-scripts and it needs the -prot-nameserver=<address>.

3. MPI_Lookup_name: service name has not been published

   This is caused by a too fast nameserver startup. This can be avoided by adding a 'sleep 5' line to the partner startup scripts, just before the eval line. Recent example scripts have this 'sleep 5' already accounted for.

9 MATLAB/Simulink coupling

MADYMO provides a coupling with MATLAB/ Simulink. The coupling is available on platforms which are supported by both MADYMO and The MathWorks®, MATLAB R2009a (see the MathWorks User guide or visit www.mathworks.com). This does not exclude possible compatibility with another MATLAB version.

This coupling is based on a TCP/IP connection between the S-function named **madymo3d_server** and MADYMO. It allows the user to run MATLAB/ Simulink on the same or different computer than where MADYMO is running. However, note that MADYMO can only connect and communicate with the same type of platform using the IP address HOST_-ADDRESS. See Fig. 9.1 for the detailed structure of the coupling of the two environments.

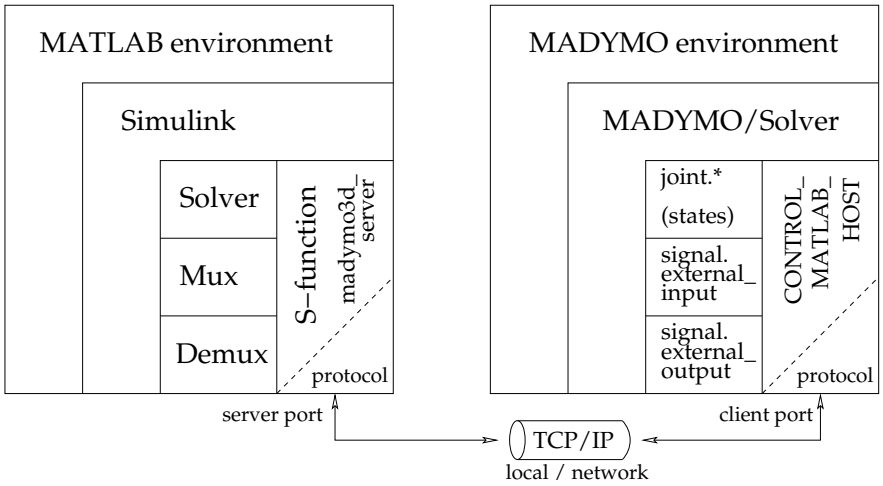


Figure 9.1: Detailed outline of the coupling.

9.1 Compatible MATLAB version

As explained, the S-function **madymo3d_server** is a coupling program running within MATLAB to communicate with MADYMO. For each program the system requirements are given by its vendor. For system requirements of the MADYMO solver see the Installation Instructions. Since we have two programmes, the system requirements may be conflicting. The coupling program of this MADYMO version is based on the MATLAB build and test versions are stated in the following table.

Solver Platformid	MATLAB build	MATLAB test
em64t-win	R2009a	R2009a
win32p	R2009a	R2009a
linux26-x86_64	R14SP3	R2009a

The coupling is tested using MATLAB version R2009a on newer platforms. Known compatibility problems are noted on the linux operating systems. Depending on the linux version the coupling either can or can not run, if this is the case a different version of the linux operating system, MADYMO or MATLAB is needed.

9.2 Interfacing MADYMO with MATLAB/ Simulink

The interface between MATLAB/ Simulink and MADYMO is defined in both the Simulink and the MADYMO model. To set up a coupled simulation, perform the following steps:

1. Link both input and output ports by defining SIGNAL.EXTERNAL_INPUT and SIGNAL.EXTERNAL_OUTPUT in the MADYMO model to Mux and Demux blocks in the Simulink model file.
2. Select the time integration method 'MATLAB'/'EULER' in the MADYMO model.
3. The simulation time and solver options in the Simulink model are selected via the Simulink graphical user interface shown in Fig. 9.2.
4. The integration time step should be set equal in both programmes.

All MADYMO functionality is available within MATLAB/ Simulink by means of a S-function called **madymo3d_server**. One or more instances of such block(s) can be defined in a Simulink model. An S-function is shown in Fig. 9.3, in which the parameters represent the following:

1. Command to be executed when the Simulink model is started. This character string normally ends with the MADYMO model XML-filename.
2. Number of 'continuous' states to be integrated by the MATLAB Solver, using INT_MTH="MATLAB" under CONTROL_ANALYSIS.TIME.
   This number should be greater than or equal to the maximum number of first order differential equations in the MADYMO model.
   Select '0' when the states are integrated by MADYMO (INT_MTH="EULER").
3. Input port dimension, largest EXTERNAL_REF defined for SIGNAL.EXTERNAL_INPUT. The value -1 represents dynamically sized.
4. Output port dimension, largest EXTERNAL_REF defined for SIGNAL.EXTERNAL_OUTPUT. The value -1 represents dynamically sized.
5. Server port number, value of HOST_PORT. MADYMO connects to this port.

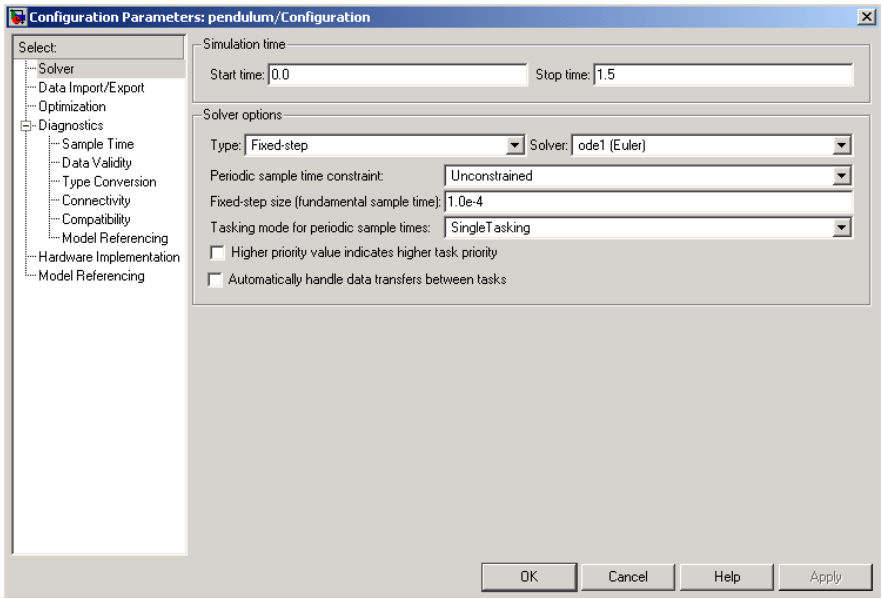


Figure 9.2: Example of the Configuration Parameters.

6. Polling time [s] for the server to wait on the connection with MADYMO.
7. Integration time step, value of TIME_STEP defined for CONTROL_ANALYSIS.TIME.

9.3 Using MATLAB/ Simulink in combination with MADYMO

Open the MADYMO environment by typing

madymo75 -sh

This is only needed when MADYMO specific environment variables are used in the m-files.

An example of the coupling, see Application Manual, is as follows:

- Open a command window (using either cmd or xterm).
- Go to the directory where the MADYMO model, the MATLAB/ Simulink model and the startup.m files are located.
- Start MATLAB from the command window.
- The file startup.m is executed automatically.

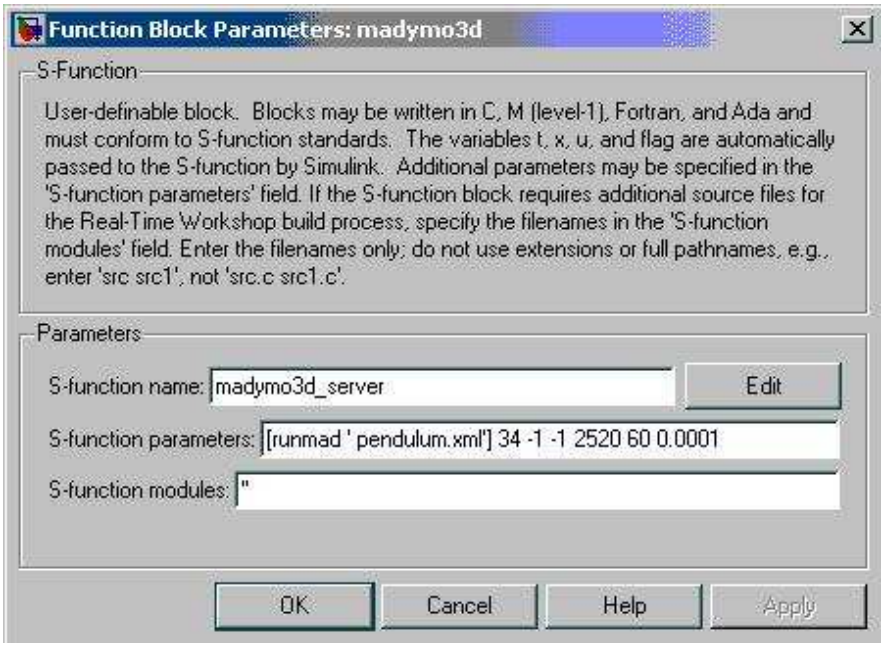

Function Block Parameters: madymo3d
S-Function
User-definable block. Blocks may be written in C, M (level-1), Fortran, and Ada and must conform to S-function standards. The variables t, x, u, and flag are automatically passed to the S-function by Simulink. Additional parameters may be specified in the 'S-function parameters' field. If the S-function block requires additional source files for the Real-Time Workshop build process, specify the filenames in the 'S-function modules' field. Enter the filenames only; do not use extensions or full pathnames, e.g., enter 'src src1', not 'src.c src1.c'.
Parameters
S-function name: madymo3d_server
Edit
S-function parameters: [runmad ' pendulum.xml'] 34 -1 -1 2520 60 0.0001
S-function modules: ''
OK
Cancel
Help
Apply

Figure 9.3: Example of the S-function settings.

- The prompt is changed into '>>' indicating that MATLAB is running.
- Load or create the associated Simulink model, for example the pendulum shown in Fig. 9.4, containing an S-function named **madymo3d_server** and appropriate parameters shown in Fig. 9.3. Here, the MATLAB variable named 'runmad' refers to the command to execute MADYMO.
- Go to Simulation ⟶ Start in the Simulink graphical user interface to start the simulation.
- Parameters or models can be altered and the simulation can be started again.
- Note that one can also edit the startup.m file by replacing the environment variable `MADHOME` which the path to the madymodir. This gives you the ability to start MATLAB by clicking on the icon. It is good practice to store the startup file in the directory where the models are located.

The coupled simulation runs until either the end time in the MADYMO model is reached, the end time in the Simulink model is reached or the simulation is terminated by the user.

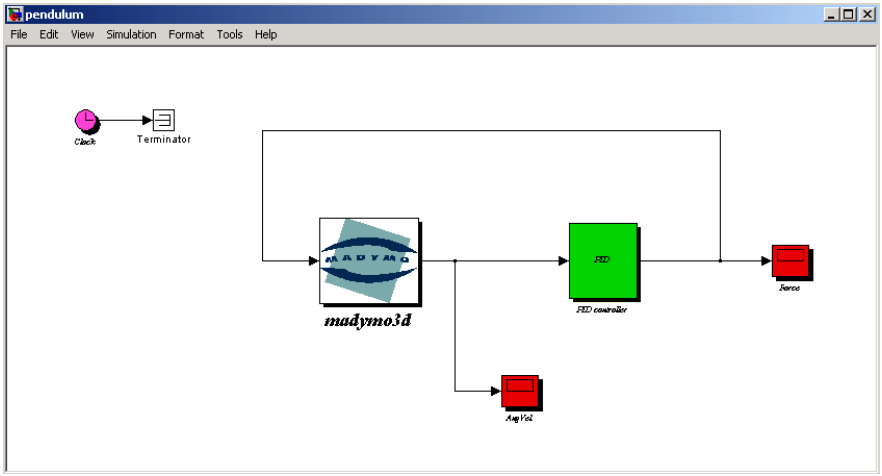


Figure 9.4: Example of the pendulum model

9.4 Conditions

Accessing MADYMO from MATLAB/ Simulink via the S-function **madymo3d_server** is straight forward if the following is taken into account:

- MATLAB executes the m-file 'startup.m' when starting. The minimum list of commands is listed below.
- The coupling can run only if the MATLAB variable runmad is set and the search path includes the S-function **madymo3d_server**, version 7.5.

9.5 Miscellaneous

Using the coupling gives some side effects, mainly in the area of numerical stability. Most of the side effects concern the availability of the time integration and the computer platforms.

Here are some MATLAB/ Simulink remarks:

- Development and testing takes place on the MATLAB Service Pack which was available at the MADYMO release date. It is most likely that newer Service Packs are working, where older Service Packs might not work on some platforms.
- Either ode1 (Euler) or ode2 (Heun) is the best MATLAB solver to start with for multi-body models. In some cases problems with ode4 are noted.
- The selection of the discrete (e.g. no continuous states) solver is the preferred choice to

start with Finite Element models. Then, the states should be computed by the MADYMO implicit-explicit Euler integrator in MADYMO.

- The variable time step MATLAB (and MADYMO) solvers are not implemented for the MATLAB/ Simulink coupling, because the integration time step is not updated in MADYMO.
- Only zero or one input/output ports are omitted. Both the input port width and the output port width should match the MADYMO model.
- During the termination phase of the Simulink model the platform 'win32p' and 'em64t-win' will continue, where the other platforms wait on the completion of the termination phase of MADYMO.
- For MATLAB running on Windows, batch processing can be used only if matlab.bat is started in combination with the option -wait. See the MATLAB manual pages for further documentation.
- The port used for the TCP/IP connection are not always available for the next simulation, since they might be locked by the system. The update time of the system administration can take quite some time.
- MADYMO information may be shown in the command screen of MATLAB during the coupled simulation. However all MADYMO related output files, including the LOG file are available separately.
- Errors in the MADYMO model will result in an error during the start of the Simulink model. An error message for the S-function block **madymo3d_server** will explain the type of error. For a detailed error message the LOG and/or REPRINT file should be examined.
- On linux, the MATLAB (environment) might cause abnormal termination during the parsing of the MADYMO input file. This is solved by setting the LD_PRELOAD environment variable of the gcc library.
  `setenv LD_PRELOAD /lib/libgcc_s.so.1`

Here are some MADYMO known issues:

- Since MATLAB leads the integration, the MB time step can not be changed during the simulation. The coupling with an external FE program is not supported.
- The stability of MADYMO is not only related to the parameters defined in the MADYMO model, but also related to the Simulink model when a closed control loop (feedback) is defined.
- Locking and unlocking by Coulomb friction, breaking of joints and mass belts are not allowed when using the MATLAB solver, because no reset of the states is implemented. The MADYMO implicit-explicit Euler integrator can be used instead.

- When connecting, MADYMO expects the **madymo3d_server** to be running on the IP address HOST_ADDRESS. It might connect to another server application when the port on the host is providing another server application. In this case the MADYMO connects successfully, but subsequently waits or terminates.
- Coupling is available for computers running in SMP mode, but not in MPP mode.
- Both MADYMO and MATLAB must be either 32-bits or 64-bits executables. Mixing 32-bit and 64-bit executables is not supported.

See MADYMO Reference Manual, section CONTROL_MATLAB_HOST, the MATLAB and the Simulink User guide for further details. TASS BV is a partner of The MathWorks.